VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

DIPLOMOVÁ PRÁCE

Brno, 2016 Bc. Štefan Zima

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

ANALÝZA GEOLOKAČNÍCH DATABÁZÍ
ANALYSIS OF GEOLOCATION DATABASES

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Štefan Zima**
AUTHOR

VEDOUCÍ PRÁCE — **doc. Ing. Dan Komosný, Ph.D.**
SUPERVISOR

BRNO 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Telekomunikační a informační technika**
Ústav telekomunikací

Student: Bc. Štefan Zima ***ID:*** 120518
Ročník: 2 ***Akademický rok:*** 2015/16

NÁZEV TÉMATU:

Analýza geolokačních databází

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s možnostmi odhadu geografické pozice IP adres pomocí geolokačních databází. Zaměřte se na geolokační databáze, které používají webové rozhraní. Zhotovte program v jazyce Python pro automatický odhad polohy zadaných IP adres pomocí komerčních geolokačních databází. Z odhadnutých poloh určete přesnost jednotlivých databází.

DOPORUČENÁ LITERATURA:

[1] PUŽMANOVÁ, R. TCP/IP v kostce. 2. vyd. Kopp, 2009. 620 s. ISBN: 978-80-7232-388-3.

[2] Linux Dokumentační projekt. 4. vyd. Computer Press, 2008. 1336 s. ISBN: 978-80-251-1525-1.

[3] POESE, I., UHLIG, S., KAAFAR, M., DONNET, B., GUEYE, B. IP Geolocation Databases: Unreliable? ACM SIGCOMM Computer Communication Review, 2011, roč. 41, č. 2, s. 53-56. ISSN: 0146-4833.

Termín zadání: 1.2.2016 ***Termín odevzdání:*** 25.5.2016

Vedoucí práce: doc. Ing. Dan Komosný, Ph.D.
Konzultant diplomové práce:

doc. Ing. Jiří Mišurec, CSc., *předseda oborové rady*

ABSTRAKT

Táto práca je zameraná na získavanie a analýzu dát z komerčných geolokačných databáz. Teoretická časť rozoberá techniky geolokácie na základe IP adries, stručne popisuje jednotlivé komerčné databázy a prístup k ich geolokačným službám. Praktická časť zahŕňa vývoj skriptov pre získavanie dát z týchto databáz a úpravou dát pre následnú analýzu. Prvý z dvojice skriptov dáta získava a ukladá ich do súboru v predpísanej forme. Druhý z dvojice skriptov upravuje získané dáta a vytvára pravidlá s pomocou ktorých je možné získavať presnejšie výsledky. Po úprave výstupných dát nasleduje podrobná analýza, ktorá sa zameriava na presnosť geolokačných služieb v niekoľkých úrovniach. Medzi tieto úrovne patrí presnosť na úrovni krajiny, regiónu, mesta, zemepisnej dĺžky a zemepisnej šírky. Výsledkom analýzy je záver, pojednávajúci o presnosti testovaných geolokačných služieb. Cieľom tejto práce je analyzovať súčasný stav geolokačných služieb komerčných databáz a vyhodnotiť ich presnosť. Použité skripty sú implementované v programovaciom jazyku Python vo verzi 3 a jeho modulov urllib, re a json. Skripty sú testované na komoditnom hardvére s operačným systémom Linux.

KĽÚČOVÉ SLOVÁ

Geolokácia, HTTP, IP, Linux, Python

ABSTRACT

This thesis is focused on collecting data from commercial geolocation databases and its analysis. The theoretical part discusses techniques of IP geolocation, briefly describes commercial geolocation databases and mechanisms for accessing their geolocation services. The practical part of this thesis involves implementation of scripts for collecting and modification data retrieved from commercial geolocation databases for further analysis. First script collects the data and store them in output files in specified format. Second script modifies gathered data to achieve better accuracy. Detailed analysis is performed afterwards. Analysis is performed at several levels. This includes country, region and city accuracy as well as accuracy on latitude and longitude parameters. Output of this analysis will be conclusive, and will also display accuracy of tested geolocation services. The aim of this thesis is to analyze currently available commercial geolocation services and to prove their accuracy. Used scripts are implemented in Python programming language in version 3 using modules urllib, re and json. Scripts are test on commodity hardware with Linux operating system.

KEYWORDS

Geolocation, HTTP, IP, Linux, Python

ZIMA, Štefan *Analýza Geolokačných Databáz*: diplomová práca. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2016. 63 s. Vedúci práce bol doc. Ing. Dan Komosný, Ph.D.

VYHLÁSENIE

Vyhlasujem, že som svoju diplomovoú prácu na tému „Analýza Geolokačných Databáz“ vypracoval(a) samostatne pod vedením vedúceho diplomovej práce, využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor(ka) uvedenej diplomovej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto diplomovej práce som neporušil(a) autorské práva tretích osôb, najmä som nezasiahol(-la) nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý(-á) následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákoníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora(-ky)

POĎAKOVANIE

Rád by som poďakoval vedúcemu diplomovej práce pánovi doc. Ing. Danu Komosnému, Ph.D. za odborné vedenie, konzultácie, trpezlivosť a podnetné návrhy k práci.

Brno ....

podpis autora(-ky)

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic
http://www.six.feec.vutbr.cz

POĎAKOVANIE

Výzkum popsaný v tejto diplomovej práci bol realizovaný v laboratóriách podporených projektom SIX; registračné číslo CZ.1.05/2.1.00/03.0072, operačný program Výzkum a vývoj pro inovace.

Brno . . . . . .

. . . . . .
podpis autora(-ky)

OBSAH

ZOZNAM OBRÁZKOV

ZOZNAM TABULIEK

ZOZNAM VÝPISOV

ÚVOD

V súčasnosti predstavuje získavanie informácií o geografickej lokácií internetového účastníka častý jav, ktorý je kľúčový pre mnoho služieb poskytovaných internetom. Tento fakt podlieha požiadavkám na prispôsobenie kontextu poskytovaných informácií pre koncových užívateľov na Internete rovnako ako aj bezpečnostným aspektom. Vzhľadom na povahu Internetu v dnešnej dobe je možné komunikovať s koncovými bodmi na opačných stranách zemegule a tým aj viesť útok v povedomí, že na útočníka nemajú právne postihy vďaka vzdialenosti dosah. Práve informácia o fyzickej polohe útočníka môže viesť k jeho dolapeniu či obmedzeniu dátového toku z oblasti so zvýšeným výskytom podobných útočných aktivít. Táto informácia môže tiež viesť k zlepšeniu prevencie pred prípadným napadnutím sieťovej infraštruktúry ale k aj iným bezpečnostným opatreniam, ako napríklad blokovaniu platobnej karty v prípade zneužitia útočníkom mimo geografickú oblasť, v ktorej sa pohybuje oficiálny vlastník. Popri značnom dôraze na bezpečnostné aspekty nemožno nespomenúť aj bežne využívané služby geolokácie internetového účastníka ako napríklad cielená reklama na danú geografickú oblasť, prispôsobenie jazykovej sady, informácie o lokálnych udalostiach a mnoho ďalších.

V súvislosti so spomenutými internetovými službami a geolokáciou, hovoríme o takzvanej IP Geolokácií. IP Geolokácia je proces získavania fyzickej polohy internetovéhó účastníka alebo zariadenia na základe IP adresy [4]. V dnešnej dobe existuje niekoľko prekážok pre získavanie tejto informácie. Jednou z nich je absencia protokolu získavajúceho informácie o polohe vybraného internetového účastníka (aj keď DNS protokol môže obsahovať túto informáciu) alebo absencia súčastí zariadenia umožňujúcich priame získanie geografickej polohy, napríklad GPS [5]. Práve pre tieto nedostatky vznikli viaceré metódy získavania informácií o polohe internetového účastníka na základe jeho IP adresy, ktoré budú predmetom Kapitoly 1.

Cieľom tejto práce je analýza existujúcich geolokačných služieb komerčných poskytovateľov a otestovanie ich v praxi. Podobná analýza bola už v minulosti predmetom viacerých pokusov, ako napríklad v práci [3], no s dynamicky sa meniacou infraštruktúrou a vývojom nových metód na získavanie geolokačných informácií je nutné neustále konfrontovať existujúce riešenia využívané predovšetkým v komerčnej sfére.

Pred samostatnou prácou s týmito službami a ich analýzou je nutné si zadefinovať niekoľko kľúčových otázok, ktoré budú zodpovedané v priebehu tejto práce po absolvovaní štúdia, rozboru teoretických poznatkov a praktických skúseností.

- Ako získať informáciu o polohe účastníka internetovej komunikácie ?
- Je získana poloha presná ?
- Ako overiť, či je získaná poloha presná ?

Teoretické zázemie tejto práce sa zameriava na definíciu a vysvetlenie princípu IP Geolokácie v Kapitole 1. Jeden zo zdrojov geolokačných informácií predstavujú geolokačné databázy, ktoré poskytujú webové alebo aplikačné rozhranie pre prístup ku svojím službám. Tieto rozhrania budú predmetom Kapitoly 2. Súčasťou tejto kapitoly je aj krátke predstavenie programovacieho jazyka Python a jeho modulov potrebných pre komunikáciu s zohraniami komerčných geolokačných databáz. Kapitola 3 rozoberá implementáciu skriptov pre získavanie dát a manipuláciu s nimi. Použitím týchto skriptov pre testovanie jednotlivých geolokačných služieb sa zaoberá Kapitola 4. Získané výsledky je nutné vopred analyzovať v Kapitole 5 a následne vyhodnotiť presnosť testovaných služieb v Kapitole 6. V poslednej, Kapitole 7, sú zrhnuté nadobudnuté poznatky.

1 IP GEOLOKÁCIA

V úvode tejto práce bolo spomenuté, že Geolokácia predstavuje proces získavania informácií o geografickej polohe skúmaného objektu. Pre účely získavania tejto informácie v rámci Internetu, je skúmaným objektom účastník alebo zariadenie, ktoré je identifikovateľné verejnou IP adresou. S využitím IP adresy a vlastností sieťovej infraštruktúry je možné získať informáciu o polohe. Tento proces je popisovaný ako IP Geolokácia alebo Geolokácia na základe IP adresy. Táto kapitola rozoberá niekoľko prístupov k realizácií tohto procesu. Podkapitola 1.1 sa zamerá na takzvané aktívne prístupy, ktoré pozostávajú z metód založených na výsledkoch aktívných meraní a topologických vlastností sieťových segmentov. Podkapitola 1.2 popisuje pasívne prístupy, ktoré predstavujú metódy získavania geolokačných záznamov z databáz. V oboch podkapitolách sú rozoberané výhody a nevýhody spomínaných metód a ich využitie v praxi na rôzne účely. Výstupom tejto kapitoly budú znalosti nutné pre formovanie požiadavkov na aplikáciu pre získavanie geolokačných záznamov a analýzu komerčných geolokačných služieb. Jednou z dôležitých otázok tohto procesu je otázka spoľahlivosti týchto služieb [3].

1.1 Aktívne techniky

Na základe technického prístupu k určovaniu geografickej polohy zariadenia s využitím informácie o jeho verejnej IP adrese sú rozlišované dve paradigmy. Prvé paradigma spočíva vo vyhodnocovaní aktívnych meraní prenosových a topologických vlastností počítačových sietí, v ktorých sa skúmané zariadenie nachádza [4]. Toto paradigma sa nazýva aktívne [3].

Príčinou tohto označenia je fakt, že informácie o polohe zariadenia sú získavané na základe série aktívych meraní a skúmaní parametrov prenosu na počítačovej sieti [7]. V prípade aktívneho prístupu k získavaní hodnôt týchto parametrov sú definované dve podmnožiny v náväznosti na ich konkrétny výber [7]. Spomenuté prenosové parametre následne slúžia k vytvoreniu podmienok pre ohodnotenie vzdialeností skúmaného zariadenia od orietačných bodov. Prenosové parametre, orientačné body a ich úloha, bude vysvetlená v nasledujúcich podkapitolách na konkrétnych metódach aktívneho prístupu. Podkapitola 1.1.1 popisuje využitie parametru oneskorenia. Podkapitola 1.1.2 sa zameriava na využitie vlastností topológií siete.

1.1.1 Metódy založené na meraniach sieťových parametrov

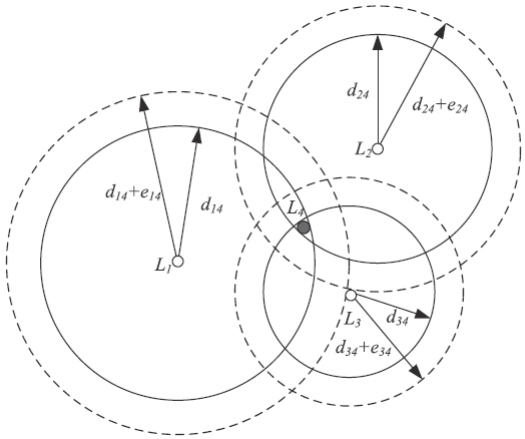
Prvou podmnožinou aktívnych techník v náväznosti na konkrétny výber parametrov sú metódy získavania polohy zariadenia na základe oneskorenia na počítačovej sieti. Dôležitou úlohou, ktorú bolo nutné vyriešiť pri pokuse o využitie tohto parametru, bola definícia mapovania oneskorenia na geografickú vzdialenosť [7]. Často využívaným parametrom [4] definujúcim oneskorenie na počítačovej sieti je RTT (Round-Trip Time) [1]. Tento parameter definuje časový úsek, ktorý musí paket (dátová jednotka 3. vrsty ISO/OSI modelu) prejsť od zdroja k cieľu a naspäť [1]. RTT parameter pozostáva z nasledujúcich častí [4]:

- Propagačného oneskorenia
- Prenosového oneskorenia
- Pracovného oneskorenia
- Oneskorenia rady

Propagačné oneskorenie je považované za deterministické. Prenosové, pracovné oneskorenie a oneskorenie rady je považované za stochastické a je určované na základe štatistických informácií [4]. Pre získavanie časového údaju parametra RTT sú používané bežné nástroje ako napríklad nástroj ping a tracert [1]. Obidve založené na protokole ICMP [10].

Pre úplny popis metódy popísanej v tejto podkapitole je nutné sa vrátiť k popisu orientačných bodov. Orientačné body slúžia na získanie hodnoty už spomínaného oneskorenia mapovaného na geografickú vzdialenosť. Tieto orientačné body majú známu geografickú polohu. Geografická poloha zariadenia s IP adresou môže byť odhadnutá prostredníctvom multilateračných techník založených na meraní oneskorenia z viacerých orientačných bodov [4]. Multilaterácia je proces určovania geografickej polohy objektu založený na rozdielnych časových príchodov signálov emitovaných z okolných orientačných bodov smerom ku skúmanému objektu [4]. Multilateráciu demonštruje obrázok 1.1 a detailnejšie popisuje nasledujúci text. Na uvedenom obrázku je rozoberaná úloha odhadu geografickej polohy zariadenia reprezentovaného premennou L_4. Orientačné body sú označené ako L_1, L_2 a L_3. Odhad vzdialenosti využíva spomenutý RTT parameter, meraný na orientačných bodoch v okolí skúmaného objektu L_4. Orientačné body vysielajú prostredníctvom protokolu ICMP dotazy na ceste smerom ku hľadanému zariadeniu a získavajú hodnoty propagačného oneskorenia pre jednotlivé cesty. Využitím funkcie mapovania, prevedú získané hodnoty oneskorenia na geografické vzdialenosti. Geografické vzdialenosti sú na obrázku 1.1 reprezentované premennými d_{14}, d_{24} a d_{34}. Aditívne oneskorenie, pozostávajúce

z prenosového, pracovného oneskorenia a oneskorenia rady, je zobrazené premennými e_{14}, e_{24} a e_{34}. Polomer plnej čiary každého z troch orientačných bodov L_1, L_2 a L_3, naznačuje spodný odhad geografickej vzdialenosti. Polomer prerušovanej čiary predstavuje horný odhad geografickej vzdialenosti. Obidva odhady boli získané s využitím parametra RTT [4]. Výsledkom tohoto merania sú kruhové oblasti, reprezentujúce možnú polohu skúmaného objektu L_4. Prienikom týchto kruhových oblastí je možné získať presnejší odhad polohy zariadenia L_4.



Obr. 1.1: Multilaterácia pre IP Geolokáciu [4]

Existuje niekoľko implementácií týchto metód [7]. Hlavným odlišujúcim sa znakom rôznych implementácií je voľba funkcie, ktorá vykonáva mapovanie parametrov oneskorenia na geografickú vzdialenosť [4]. Medzi tieto metódy patrí napríklad IP2Geo alebo CBG (Constrait-Based Geolocation) [7].

1.1.2 Metódy založené na topologických vlastnostiach

Druhou podmnožinou aktívnych techník v náväznosti na konkrétny výber parametrov sú metódy získavania polohy zariadenia na základe vlastností sieťovej topológie. Tieto metódy kombinujú využitie parametru oneskorenia spolu s informáciami o topológii pre dosiahnutie vyššiej presnosti [7]. Informácie o topologických vlastnostiach sietí sú získavané pri prechode paketov cez aktívne sieťové prvky a zaznamenávaním dostupných informácií z ciest po vybraných linkách. Do týchto informácií spadajú prevažne informácie od poskytovateľov pripojenia v danej oblasti [7]. Jednou z aplikácií metódy založenej na topologických vlastnostiach je TBG (Topology-Based Geolocation) [7]. Táto aplikácia využíva odhad geografickej polohy zariadení na Internete na základe obmedzujúcich podmienok, ktoré sú kombináciou hodnôt oneskorenia s topologickými informáciami.

Aktívne techniky získavania geografickej polohy zariadení v Internete nie vždy pracujú spoľahlivo. V určitých podmienkach môže orientačný bod vykazovať hodnoty oneskorenia výrazne odlišujúce sa od predpokladanej hodnoty, čím zamedzuje korektnému vyhodnoteniu polohy. Ďalším problémom je orientačný bod úmyselne meniaci parameter RTT na hodnoty, ktoré neopovedajú skutočnosti, za účelom zamedzenia geolokácie.

V praxi sa vyskytujú hybridné modely kombinujúce obidva technické prístupy za účelom zvýšenia presnosti [4]. Aktívne techniky prinášajú malú škálovateľnosť, veľkú záťaž na spracovanie a značnú časovú náročnosť pohybujúcu sa od desiatok sekúnd až po niekoľko minút pre geolokáciu jedinej IP adresy [3]. Tento fakt vyplýva z povahy meraní a aktívneho prístupu k získavaniu informácií zo siete. Aj napriek tomu, že sú tieto metódy náročné na spracovanie, môžu poskytovať výrazne vyššiu presnosť oproti pasívnym metódam. Pasívne metódy sú predmetom nasledujúcej podkapitoly 1.2.

1.2 Pasívne techniky

Druhý technický prístup k určovaniu polohy zariadenia s využitím informácie o jeho IP adrese využíva databázový prístup k existujúcim geolokačným záznamom. Tento prístup alebo paradigma je označované ako pasívne [3].

Označenie pasívne tieto techniky získali preto, že záznam o geografickej polohe danej IP adresy môže existovať a nie je nutné vykonávať opakovaný výpočet ako u spomenutých aktívnych techník. Tento geolokačný záznam je načastejšie uložený v databázach poskytovateľov geolokačných služieb. Budovanie týchto databáz patrí medziu jednu z najrozšírenejších techník pri poskytovaní geolokačných služieb [3]. Ukladané záznamy reprezentujú mapovanie IP adresových blokov na geografickú polohu. V súčastnosti existujú dve formy týchto databáz z hľadiska prístupu k záznamom a služieb, ktoré poskytujú. Jedná sa o komerčné a voľne dostupné geolokačné databázy.

Komerčné geolokačné databázy poskytujú svoje služby na základe licenčných podmienok, akceptovaných pri regitrácii alebo zákupeniu geolokačnej služby. Voľne dostupné databázy poskytujú svoje služby zadarmo, avšak je otázna ich spoľahlivosť a niekedy aj dustupnosť služieb, ktorá nie je vždy garantovaná. Vzľadom na využitie IP Geolokácie pre komerčné účely je nutné zaistiť spomínanú dostupnosť služieb a ich kvalitu.

Existuje niekoľko aspektov podporujúcich fakt, že geolokačné databázy nie sú tak spoľahlivé a presné, ako su prezentované. Prvým aspektom je to, že navzdory rozsiahlym počtom záznamov, ktoré môžu obsahovať, je pokrytá len časť územia kontinentov, predstavujúca známejšiu a vyspelejšiu oblasť, ako je napríklad USA[3]. Tento poznatok prináša nerovnováhu v reprezentácii IP adresových blokov a ich mapovaniu na konkrétne krajiny. Pri mapovananí IP adresových blokov na geografické celky je tiež možné použiť smerovacie protokoly ako napríklad BGP [11]. Použitím tohoto prístupu pri budovaní databáz geolokačných záznamov môže dôjsť k nedostatkom, ktoré vedú k výraznej miere odlišnosti skutočného mapovania IP adresových blokov na správne geografické lokácie [11]. Napriek spomínaným nedostatkom je možné vysloviť tvrdenie, že geolokačné databázy môžu podporovať úspešný odhad na úrovni krajín avšak rozhodnie nie ja úrovni regiónov alebo miest [3].

Geolokačné databázy sú postavené na rôznych databázových modeloch a technológiach akými sú napríklad SQL, MongoDB a podobne [12]. Prístup k jednotlivým záznamom je vykonávaný cez webové a aplikačné rozhranie, poskytované správcami jednotlivých databáz. Typický záznam geolokačnej databázy obsahuje pár korešpondujúci s celočíselnou reprezentáciou minima a maxima IP adresového bloku. Každý z týchto blokov je následne asociovaný s ďalšími užitočnými informáciami akými sú napríklad názov krajiny, regiónu, mesta, zemepisná šírka a zemepisná dĺžka [3].

Správcovia geolokačných databáz umožňujú dva druhy prístupu ku geolokačným službám. Voľne dostupné databázy poskytujú možnosť využitia geolokačných záznamov v plnej miere a neobmedzenom množstve avšak so spomínanou mierou nepresností a nedostatkov, ktoré znemožňujú správny odhad polohy. Vzhľadom na veľký počet záznamov je nutné udržovať istú mieru konzistencie a zaisťovať úplnosť a aktuálnosť, čo je u voľne dostupných databáz často diskutabilné [3]. U komerčných geolokačných databáz nie je vždy tento problém výrazne menší. Navyše pri komerčnej aplikácií nie je môžné nahliadnúť do vnútornej metodológie získavania geolokačných záznamov. Z povahy prideľovania IP adries od ISP môžeme predpokladať, že obnova záznamov voľných či komerčných geolokačných databáz nie je častá a preto je nutné neustále konfrontovať ich presnosť a uskutočniť vlastnú analýzu.

Odhad presnej polohy zariadenia s IP adresou je kľúčovým aspektom geolokačných databáz a predovšetkým u platených geolokačných služieb. Predstavuje garanciu, ktorou je možné zaistiť dostupnosť služieb využívajúcich IP geolokáciu a záujem zo strany verejnosti. Predmetom tejto práce sú komerčne geolokačné databázy, ktoré poskytujú široké portfólio geolokačných služieb na viacerých úrovniach. Medzi tieto databázy patria MaxMind [13], DB-IP [14], IP2Location [15], IPInfo [16], Skyhook [17], EurekAPI [18] a Geobytes[19]. Prehľad prezentovaných presností a možností týchto databáz prezentuje tabuľka 1.1. V prípade neznámej hodnoty je v tabuľke uvedená hodnota `N`.

Tab. 1.1: Presnosť a pokrytie prezentované komerčnými databázami

Databáza	Krajina[%]	Mesto[%]	IP
MaxMind	99.8	81 (<50 km)	100%
DB-IP	N	N	9251417
IP2Location	99.5	> 75	N
IPInfo	N	N	N
Skyhook	N	N	N
EurekAPI	N	N	4294967296
Geobytes	97	80.25 (< 100km)	100%

2 ROZHRANIA GEOLOKAČNÝCH DATABÁZ

Spomínané komerčné geolokačné databázy v predchádzajucej kapitole poskytujú dve základné metódy prístupu ku geolokačným službám a jednotlivým geolokačným záznamom. Prvá z metód využíva webové rozhrania a elementy formulárov, ktorými sa po odoslaní špecifikuje požiadavok na databázu s geolokačnými záznamami vybranej IP adresy. Ako odpoveď je získaná webová stránka v HTML kóde [8], ktorá obsahuje vložený geolokačný záznam napríklad v podobe tabuľky. Spracovaním tohto kódu získame požadované informácie, ktoré môžeme použiť na ďalšie spracovanie. Druhou metódou získavania geolokačných záznamov je použitím aplikačného rozhrania databáz. Odpoveďou v tomto prípade predstavuje reťazec znakov vo formáte JSON [22]. Obidve prístupy využívajú metódy protokolu `HTTP`, metódu `GET` a `POST`. Podkapitola 2.1 sa zameriava na prácu s týmito metódami v náväznosti na webové formuláre, ktoré tvoria hlavný komunikačný kanál.

Interpretované programovacie jazyky nám umožňujú vytvoriť jednoduchú aplikáciu na prácu s týmito databázami, postavenú na tvorbe a odosielaní požiadavkov protokolu `HTTP`. Prácu s webovými a aplikačnými rozhraniami týchto databáz je možné jednoducho automatizovať a docieliť tak jednoduchý, učinný a rýchly spôsob pristúpu ku geolokačným službám komerčných geolokačných databáz. Jedným z programovacích jazykov, ktorý umožňuje splniť tieto požiadavky, je čoraz viac populárny programovací jazyk Python. V podkapitole 2.2 bude v krátkosti predstavený, rovnako ako aj jeho moduly potrebné pre prácu s rozhraniami databáz.

2.1 Komunikácia s rozhraním

Povahou webových geolokačných služieb je sprostredkovať geolokačný záznam na základe požiadavku na databázu, ktorej realizácia je vnútorne skrytá. Aby bolo možné komunikovať s touto databázou a vytvárať konkrétne požiadavky, je nutné definovať rozhranie. V prvom prípade sa o zostavenie požiadavku stará webový formulár a jeho polia, ktorý po odoslaní vytvára dotaz na databázu a konkrétny záznam. Odpoveďou je informácia zakódovaná v jazyku HTML. U aplikačného rozhrania je vytvorený konkrétny dotaz pomocou hlavičiek protokolu `HTTP` a údajoch zakódovaných do URL adresy geolokoačnej databázy.

V obidvoch vyššie spomenutých prípadoch sa pracuje s protokolom `HTTP` a jeho metódami `GET` a `POST`. Tieto metódy sú detailne popísané v dokument [21]. V tejto podkapitole je rozoberané konkrétne využitie na príklade webových formulároch. Ich vnútorná realizácia a konkrétna implementácia nie je pre túto prácu predmetom skúmania.

2.1.1 Formuláre

Formuláre sa skladajú z dvoch hlavných častí [8]. Prvá z nich je tvorená skupinou polí, predstavujúcich napríklad textové polia, tlačítka, zoznamy s možnosťami výberu a podobne. Druhou dôležitou časťou sú skripty na strane servera, ktoré spracovávajú vstup od užívateľa, kontrolujú správnosť formuláru a vytvárajú požiadavok na databázu. Štandardným programovacím jazykom pre tieto účely je jazyk PHP [8]. Vzhľadom na to, že sa jedná spracovanie na strane servera, je v kompetencii správcu geolokačnej databázy tieto skripty vytvárat a spravovať. Formulár je tvorený troma dôležitými súčasťami [8]:

- Element `form`, ktorý obsahuje atribúty `action` a `method`.
- Formulárové polia, napríklad element `input` predstavujúci textové pole.
- Tlačítko na odosielanie formuláru.

Atribút `action` predstavuje ako už anglický názov napovedá, akciu na stisk tlačítka pre odoslanie dát formulárových polí na ďalšie spracovanie. Toto spracovanie je ako už bolo povedané, vykonávane na strane servera PHP skriptom. Tento skript následne vytvorí požiadavok na databázu, ktorý následne odošle a získa odpoveď vo forme použitého databázového modelu. Výsledok je transformovaný do kódu jazyka HTML a výsledok je zobrazený užívateľovi. Na konkrétnu formu kódu s geolokačnými informáciamu zohľadnuje podkapitola 3.1.3, ktorá sa bude zaoberať návrhom aplikácie v jazyku Python. Zameria sa na formulárové HTML kódy komerčných poskytovateľov geolokačných služieb ako aj ich odpoveďami pre ďalšie spracovanie.

Ďalším dôležitým atribútom elementu `form` je atribút `method`. Tento atribút popisuje výber metódy protokolu `HTTP`, ktorou sa odošlú dáta formulárových polí. V prípade IP Geolokácie sa jedná o textové pole s IP adresou hľadanej stanice. Výber metódy môže byť ovplyvnený povahou formulára a to či sa jedná o citlivé dáta zadávané postupne do jeho polí alebo nie. Dotaz na informáciu o polohe dotazovanej IP adresy nie je považovaný za citlivú informáciu avšak výber metódy nemusí podliehať tomuto faktu. Pre citlívé dáta je typické využitie metódy `POST` vzhľadom

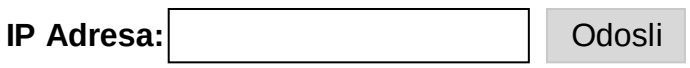
na to, že formulárové dáta sú prenášané v tele tejto metódy a nie v URL adrese, ako je to u metódy `GET`. Napriek bezpečnostným aspektom je častejšie využívaná metóda `POST` [8]. Dôvodom je možnosť prenášať väčšie množstvo dát [8].

Aby bolo možné vložiť informáciu o IP adrese, je nutné definovať formulárové pole, ktoré túto informáciu bude niesť. Týmto formulárovým poľom môže byť napríklad pole vytvorené HTML elementom `input` s atribútom `type`, ktorý definuje typ vstupných dát. Pre textovú formu je volená hodnota `"text"`. Pre IP adresu neexistuje hodnota atribútu type, preto je nutné overiť správnosť zápisu v tomto poli prostredníctvom skriptu na strane webového klienta alebo na strane serveru. Odoslanie týchto dát je realizované prostredníctvom akcie vyvolanej po stisku tláčika. Akcia zodpovedá prideleniu dát formulárových polí PHP skriptu, ktorý je definovaný v atribúte `action` elementu `form`. Príklad HTML kódu jednoduchého formuláru s opísanými základnými prvkami je vo výpise 2.1.

Výpis 2.1: Jednoduchý HTML formulár

```
<form action="ziskajpolohu.php" method="get">
IP Adresa:
  <input type="text" name="ipadresa">
  <input type="submit" value="Odosli">
</form>\
```

Takto vytvorený formulár sa bez formátovania kaskádovými štýlmi [8] zobrazí ako na obrázku 2.1. Tento krátky formulár predstavuje textové pole, do ktorého užívateľ zadá IP adresu zariadenia. Po stlačení tlačítka `Odosli` sú dáta z tohto poľa predané skriptu `ziskajpolohu.php`, ktorý vytvorí dotaz na databázu a vráti výsledok opäť v podobe webovej stránky v jazyku HTML. Ako už bolo niekoľkokrát zdôraznené, skript vykonávajúci spracovanie formulárových polí a vytváranie dotazu na databázovú službu je spravovaný správcom geolokačnej databázy.



Obr. 2.1: HTML formulár

Akcia pridelená odoslaním tohto formuláru využíva metódu `GET`. IP adresu formulárového poľa `input` v tomto prípade môžeme definovať ako voliteľný parameter URL adresy *http://ziskajpolohu.php?ipadresa=192.168.1.1*.

2.1.2 Aplikačné rozhranie

Komerčné Geolokačné databázy často poskytujú aplikačné rozhranie pre zjednodušenie prístupu ku geolokačným službám. Toto aplikačné rozhranie sa nazýva RESTové alebo zjednodušene REST API [23]. REST predstavuje architektúru aplikačného rozhrania, ktorá používa bežné metódy protokolu `HTTP`. Medzi najčastejšie používané metódy patria napríklad už spomínané `GET`, `POSTP` a `PUT` a `DELETE`.Odpoveď býva narozdiel od stránky kódovanej v jazyku HTML, kódovaná v jazyku XML alebo JSON [22], ktoré sú jednoduchšie na spracovanie vo vlastnej aplikácií komunikujúcej so serverom. Účelom jednotlivých metód je:

- `GET` - Získanie záznamu z databázy.
- `PUT` - Vytvoriť nový záznam.
- `DELETE` - Zmazať existujúci záznam.
- `POST` - Upraviť existujúci záznam alebo získanie dát z databázy.

U metódy `POST` existuje viacero spôsobov využitia. U spomenutého získania dát môže dotaz na REST rozhranie vykonať autentifikáciu užívateľa, ktorá je nutná v prípade, že sa jedná o platené služby komerčných databáz. Ta sa obvykle vykonáva na základe overenia hodnoty kľúča v prenášanom poli v tele dotazu. Príklad dotazu metódou `GET` pre zisk geolokačných informácií môže vyzerať nasledujúco:

GET www.maxmind.com/geoip/v2.1/city/130.32.34.153?demo=1 HTTP/1.1

Dotazom na REST aplikačné rozhranie je získaná odpoveď vo formáte JSON [22]. Opoveď na dotaz obsahuje geolokačné informácie dotazovanej IP adresy vo výpise 2.2. Výpis je sktrátený z dôvodu šetrenia miestom.

Výpis 2.2: Odpoveď od servera dotazovaním sa na REST aplikačné rozhranie

```
...
 "location": {
  "latitude": 51.5144,
  "longitude": -0.0941,
  "time_zone": "Europe\/London"
},
...
```

V podkapitolách 2.1.1 a 2.1.2 boli popísané metódy protokolu `HTTP` a ich využitie pri komunikácií s webovým a aplikačným rozhraním. V oboch prípadoch je definovaný požiadavok na správcu geolokačnej databázy. Typickým prístupom vlastníkov komerčných geolokačných databáz je poskytovanie oboch rozhraní, pre overenie rýchlej dostupnosti služby a jednoduchosti implementácie aplikácie na prácu s geolokačnou službou. Nasledujúca podkapitola 2.2 sa zameriava na možnosti programovacieho jazyka Python pri tvorbe týchto aplikácií.

2.2 Využitie jazyka Python

Python je interepretovaný multiplatformový programovací jazyk, ktorý je v súčasnosti jeden z najpoužívanejších jazykov vďaka svojej jednoduchosti osvojenia si základov a riešenia komplexných úloh na krátkom úseku kódu [9]. Má silnú expresivitu a poskytuje výbornú prenostiteľnosť na rôzne platformy bez nutnosti kompilácie kódu. Tento fakt podporujú vlastnosti jeho štandardnej knižnice, ktorá je plne transparentná a rovnako multiplatformová [9]. Štandardná knižnica jazyka Python poskytuje viaceré moduly, ktoré zjednodušujú prácu pri vytváraní klientských aplikácií komunikujúcich s webovými službami. Medzi tieto moduly patria, modul *urllib*, *re* a modul *json*. Všetky tieto moduly budú rozoberané v nasledujúcich podkapitolách.

2.2.1 Modul urllib

Výber programovacieho jazyka pre aplikáciu ovplyvňujú uvedené skutočností o jednoduchosti osvojenia si základov a tiež dobrej čitateľnosti kódu. Syntax jazyka Python priamo podporuje úhľadný a pre ľudské oko vhodne formátovaný kód čo patrí medzi jeho nesporné výhody [9]. Tento spôsob zápisu kódu výrazne zjednodušuje pochopenie knižnice *urllib* potrebnej pre vytvorenie `HTTP` dotazov na webové či aplikačné rozhranie geolokačných databáz.

Knižnica *urllib* umožňuje automatizovať prácu s webovými službami ďefinovaním dotazov protokolu `HTTP` a ich následným odosielaním. Dotazy su potrebné pre špecifikáciu požiadavku na IP Geolokáciu zadanej IP adresy. Ako bolo popísané v podkapitolách 2.1.1 a 2.1.2, dotazovaná IP adresa je prenášaná v URL adrese alebo v niektorom z polí v tele dotazu. V oboch prípadoch je IP adresa zadávaná do formulárového poľa alebo ako parameter definovaný REST aplikačným rozhraním. V prvom prípade sa dotaz odkazuje na atribút `name` formulárového poľa, elementu `input`. Vo výpise 2.1 v podkapitole 2.1.1 má tento atribút hodnotu *ipadresa*. Rovnakým spôsobom môže byť aplikačným rozhraním definovaný parameter `name`, ktorý nesie informáciu o požadovanej IP adrese. Po získaní tychto informácií je možné zostaviť

dotaz protokolu `HTTP` využívajúci niektorú z metód. Vo výpise 2.3 je príklad vytvorenia dát, ktoré obsahujú názov vstupného formulárového poľa s hodnotou IP adresy. Tá je zakódovaná funkciou knižnice *urllib*, funkciou `urlencode()` do URL adresy, použitej pri odosielaní dotazu. Následne je vytvorné hlavička dotazu protokolu `HTTP` v premennej *headers*, ktorá definuje užívateľského klienta. Toho maskujeme za prehliadač. Dáta obsahujúce IP adresu sú v poslednom príkaze zakódované do znakovej sady `UTF-8`.

Výpis 2.3: Dáta pre vytvorenie HTTP dotazu

```
data = {'ipadresa' : '192.168.1.1'}
data = urllib.parse.urlencode(data)
headers = {'User-Agent' : 'Mozilla/5.0'}
data = data.encode('utf-8')
```

Takto pripravené dáta slúžia na vytvorenie objektu dotazu `GET` zavolaním konštruktoru [9] objektu `Request`. Objekt dotazu je odoslaný funkciou `urlopen()`. Odpoveď od servera je uložená v objekte `reponse`. Zavolaním metódy `read()` na týmto objektom sú vrátené dáta, ktoré je však nutné vopred dekódovať funkciou `decode()` s parametrom `utf-8`. Príklad kódu realizujúci tento proces je vo výpise 2.4.

Výpis 2.4: Vytvorenie a odoslanie HTTP dotazu

```
request = urllib.request.Request(url, data, headers)
response = urllib.request.urlopen(request)
responseData = response.read().decode('utf-8')
```

V tejto podkapitole bolo predstavené zostavenie a odoslanie `HTTP` dotazu na webový server volaním metód knižnice *urllib*. Opoveď od servera predstavuje postupnosť bajtov, ktorú je nutné dekódovať a získať z nej požadované informácie. Odpoveď môže predstavovať kód v jazyku HTML [8] alebo v prípade aplikačného rozhrania dáta vo formáte JSON [22]. Spracovaním odpovedí v týchto formátoch sa zaoberajú nasledujúce podkapitoly.

2.2.2 Modul re

Pre analýzu obsahu webových stránok kódovaných v jazyku HTML sa naskytuje viacero možností. Medzi ne patrí použitie veľkého množstva externých modulov dostupných pre viaceré verzie interpretu jazyka Python, pracujúce s elementárnou štruktúrou jazyka HTML alebo na úrovni dátového prúdu bajtov [8].

Pri rozhodnutí použiť prvý zo spomínaných prístupov, je nutné zohľadniť časovú náročnosť pri štúdiu aplikačného rozhrania, prispôsobenie vstupu a výstupu použitého externého modulu. Vzhľadom na fakt, že externé moduly nie sú súčasťou štandardnej distribúcie Pythonu, je nutné sa pripraviť na možnosť straty podpory a tak k nemožnosti ďalšej aktualizácie aplikácie z dôvodu závislosti na použitej verzii.

Vo vlastnom spracovaní odpadáva časová náročnosť pri štúdiu externého modulu avšak prináša potrebu dostatočnej znalosti spracovania samostatného kódu jazyka HTML dostupnými technikami a modulmi štandardnej knižnice. Medzi tieto moduly patrí modul *re* [9], ktorý poskytuje aplikačné rozhranie pre prácu s regulárnymi výrazmi. Vysvetlenie logiky regulárnych výrazov nie je predmetom tejto práce. Pre krátky úvod do problematiky je možné použiť zdroj [9].

Použitie regulárnych výrazov predstavuje priamočiary spôsob získavania požadovaných informácií zakódovaných v jazyku HTML. Geolokačné dáta sú zakódované v ucelenej štruktúre, ktorú vo väčšine prípadov predstavuje napríklad tabuľka. Tá je tvorená skupinou párových elementov `tr` a `td`. Pre získanie obsahu týchto elementov je nutné vytvoriť regulárny výraz, ktorý spracuje odpoveď od servera a nájde zhody regulárneho výrazu s podreťazcami v prehľadávaných textových dátach.

Hľadaný reťazec je vyhľadávany metódou `re.search()`, ktorá prijíma tri parametre. Prvým parametrom je regulárny výraz vo formáte `r"vyraz"`. Pred priamým použitím je možné regulárny výraz vopred preložiť, vložiť do objektu a nad týmto objektom zavolať spomínanú metódu `search()`. Preklad regulárneho výraz vyvolá metóda `re.compile()`. Druhým parametrom metódy `search()` je reťazec, v ktorom je hľadaná zhoda s regulárnym výrazom. Pre potreby získania geolokačným informácií predstavuje tento reťazec vrátený HTML dokument. Po úspešnom nájdení podreťazca odpovedajúcemu regulárnemu výrazu, vráti metóda `search` objekt zhody alebo hodnotu `None` v prípade nezhody. Objekt osahuje metódu `group()`, ktorá prijíma číselný parameter odkazujúci na konkrétnu časť zhody s regulárnym výrazom. Nulový parameter ukazuje na celý podreťazec odpovedajúci regulárnemu výrazu. Ďalšie číselné hodnoty poukazujú na jednotlivé uzátvorkováné skupiny znakov v regulárnych výrazoch. Jednoduchý príklad demonštruje výpis 2.5.

Výpis 2.5: Práca s regulárnymi výrazmi v jazyku Python

```
>>> import re
>>> regularny_vyraz = re.compile(r"[abc]+")
>>> objekt = regularny_vyraz.search("ab")
>>> print(objekt.group(0))
ab
>>> objekt = regularny_vyraz.search("dab")
>>> print(objekt.group(0))
ab
>>> objekt = regularny_vyraz.search("dd")
>>> print(objekt.group(0))
Traceback (most recent call last):
File "<pyshell#9>", line 1, in <module>
print(objekt.group())
AttributeError: 'NoneType' object has no attribute 'group'
```

Výpis 2.5 ukazuje použitie interaktívneho vývojového prostredia `IDLE`, ktoré je štandardnou distribúciou jazyka Python [9]. Umožňuje jednoduché otestovanie jazykových konštrukcií pred samostatným použítim v konečnom programe.

2.2.3 Modul json

Pri dotaze na aplikačné rozhranie geolokačných databáz je odpoveďou dokument obsahujúci dáta vo formáte JSON [22]. Pre pohodlnú prácu s týmto formátom poskytuje programovací jazyk Python modul *json*, ktorý umožňuje jeho kódovanie a dekódovanie. Po obdržaní odpovede zo servera je nutné ju vopred dekódovať pre detekciu postupnosti znakov, napríklad znakovej sady unicode. Takto pripravené dáta obsahujúce reťazec vo formátovaní JSON sú následne prevedené do dátovej štruktúry reprezentujúcej slovník pre jednoduchý prístup ku konkrétnym častiam. Nad objektom slovníku sú volané metódy prístupu k jednotlivým úrovniam zanorenia a samotným informáciam. Jedno z týchto metód je napríklád metóda `dict.get()`, ktorá vracia hodnotu reprezentovanú kľúčom ako jej parametrom. Všetky tieto kroky prezentuje výpis 2.6.

Výpis 2.6: Spracovanie JSON v jazyku Python

```
>>> import json
>>> jsonData = '{"latitude":20}'
>>> data = json.loads(data)
>>> data.get('latitude')
20
```

V predošlých podkapitolách boli rozobrané dôležité časti rozhraní geolokačných databáz a možnosti prístupu k nim. V podkapitolách rozoberajúcich jazyk Python a jeho moduly boli ukázané spôsoby, akými je možné využiť silu tohto jazyka pre zostavenie aplikácie komunikujúcej s týmito rozhraniami. Nasledujúca kapitola je zameraná na tvorbu tejto aplikácie a aplikácie, ktorá pracuje s výstupnými dátami z týchto databáz v rôznych formátoch.

3 VÝVOJ APLIKÁCIÍ

Po získani teoretického základu pre prácu s jazykom Python a jeho modulmi je možné prístúpiť k návrhu a implementácií aplikácií pre komunikáciu s rozhraniami geolokačných databáz a úprave výsledných dát. Prvej aplikácií, ktorá komunikuje s rozhraniami geolokačnými informáciami sa venuje podkapitola 3.1. Rozoberá špecifikáciu, návrh a implementáciu kľúčových častí aplikácie v náväznosti na predošlú Kapitolu 3. S ohľadom na nadobodnuté teoretické poznatky možno v tomto prípade hovoriť o pasívnej technike získavania geolokačných informácií. Druhá podkapitola 3.2 sa zameriava na aplikáciu pre tvorbu pravidiel pri spracovaní výstupných dát. Opakovaným získavaním geolokačných informácií pri použití spomínaných pravidiel možno docieliť zlepšenie štatistických výsledkov jednotlivých databáz. Podstatnou časťou vývoja bola aplikácia v podkapitole 3.1, ktorá dáta získava, preto je jej venovaná väčšia časť Kapitoly 3.

3.1 Získavanie geolokačných záznamov

Každý vývoj softwéru podlieha niekoľkým fázam [24]. V prípade vývoja aplikácie pre získavanie geolokačných záznamov je to fáza špecifikácie požiadavkov, ktorej je venovaná podkapitola 3.1.1, fáza analýzy požiadavkov v podkapitole 3.1.2, fáza návrhu aplikácie v podkapitole 3.1.3 a samostatná implementácia v podkapitole 3.1.4. Každá z fáz sa zameriava na rôzne aspekty aplikácie a pripravuje základ pre nadchádzajúce fázy vývoja. Vstupným bodom je špecifikácia požiadavkov na aplikáciu, ktoré je nutné podrobne analyzovať. Výstupom analýzy je potom ucelený návrh kľúčových častí aplikácie a jednotlivých modulov. Po preskúmaní návrhu je možné pristúpiť k implementácii aplikácie.

3.1.1 Špecifikácia požiadavkov

Všeobecným cieľom aplikácie je zisk geolokačným informácií v závislosti na vstupe dát od užívateľa. Ako už úvodná veta napovedá, je nutné špecifikovať vstupy, spracovanie a výstupy aplikácie. Vstup aplikácie predstavuje súbor záznamov na jednotlivých riadkoch. Formát záznamu je definovaný ako riadok obsahujúci polia reťazcov, oddelených špeciálnym prázdnym znakom. Prázdnym znakom alebo inač povedané oddelovačom môže byť medzera alebo tabulátor.

Vstupný záznam obsahuje informácie na spracovanie a vytvorenie dotazu na geolokačné databázy. Polia jednotlivých záznamov predstavujú nasledujúce informácie:

1. `id` - Unikátny identifikátor stanice.
2. `ip` - IPv4 adresa stanice.
3. `dns` - Lokálny DNS server.
4. `continent` - Kontinent.
5. `country` - Krajina.
6. `region` - Región.
7. `city` - Mesto.
8. `unknown` - Nevyužité pole pre kompatibilitu.
9. `unknown` - Nevyužité pole pre kompatibilitu.
10. `latitude` - Zemepisná šírka.
11. `longitude` - Zemepisná dĺžka.
12. `unknwon` - Nevyužité pole pre kompatibilitu.

Spomedzi všetkých týchto 12 polí je najdôležitejším práve pole obsahujúce informáciu o IPv4 adrese stanice. Toto pole je použité na vytvorenie dotazu na geolokačnú databázu, ktorá vyhodnotí polohu požadovanej stanice a vrátí výsledky na ďalšie spracovanie. Medzi testované geolokčné databázy patria už spomínané:

- MaxMind [14].
- DB-IP [15].
- IP2Location [16].
- ipInfo [17].
- Skyhook [18].
- EurekAPI [19].
- Geobytes [20].

Po získaní odpovedí z týchto databáz je potrebné extrahovať informácie dôležité pre porovnávanie so vstupnými dátami. Po získaní týchto informácií, je na nich možné aplikovať pravidlá pre nahradenie a odstránie, ktoré upresnia výslednú polohu stanice na viacerých úrovniach.Výtváranie pravidiel výchádza z prvotnej analýzy získaných výsledkov pomocou skriptu popisovaného v podkapitole 3.2. Po voliteľnej aplikácií pravidiel sú záznamy porovnávané a výsledky porovnávania sú zapísané do výstupného súboru.

Výstupný súbor obsahuje všetky polia záznamu vo vstupnom súbore a výstupné dáta získaných informácií oddelené od vstupných názvom použitej geolokačnej databázy. Výstupné polia obsahujú:

13. `database` - Názov použitej databázy.
14. `country estimation` - Krajina získaná z databázy.
15. `country match` - Výsledok porovnávania na úrovni krajiny.
16. `region estimation` - Región získaný z databázy.
17. `region match` - Výsledok porovnávania na úrovni regiónu.
18. `city estimation` - Mesto ziskané z databázy.
19. `city match` - Výsledok porovnávania na úrovni mesta.
20. `latitude estimation` - Zemepisná šírka získaná z databázy.
21. `longitude estimation` - Zemepisná dĺžka získaná z databázy.
22. `error` - Chyba odhadu v kilometroch.

Dôvodom zobrazené číslovanie je fakt, že výstupný záznam obsahuje tiež všetky polia vstupného záznamu. Všetky polia výstupného záznamu sú opäť oddelené medzerou alebo tabulátorom. V prípade, že databáza vráti akékoľvek prázdne pole, tak je vo výstupe nahradené znakom pomĺčky. Výsledky zhody jednotlivých polí sú označované reťazcami *YES*, *NO* alebo v prípade neznámeho záznamu *UNK*.

V prípade že vstupné záznamy obsahujú nesprávny počet polí, je úživateľ na tento fakt upozornený výpisom o chybnom tvare vstupných dát. Táto informácia je súčasťou výstupu do súboru s tým rozdielom, že namiesto polí 14 až 22 obsahuje reťazec znakov reprezentujúci informáciu o chybe formátu vstupného záznamu a spracovanie pokračuje dalším záznamom.

Názov výstupného súboru je generovaný automaticky prostredníctom reťazca identifikujúceho použitú databázu a časového údaju vo formáte `ISO 8601`. Použité identifikačné reťazce zahrňujú:

- maxMindGeoIp2Pre - Databáza MaxMind [14].
- dbIpToLoc - Databáza DB-IP [15].
- ip2LocDb24 - Databáza IP2Location [16].
- ipInfo - Databáza IPInfo [17].
- skyhookHyperlocal - Databáza Skyhook [18].
- eurekAPI - Databáza EurekAPI [19].
- geobytes - Databáza Geobytes [20].

Oddeľovačom názvu databázy a častového údaju je podtrhovník. Celý názov výstupného súboru má potom nasledujúci formát:

názovDatabázy_dátumAčas.dat

Oddeľovače vstupných a výstupných záznamov sú voliteľné užívateľom. Ten ich volí na základe parametrov príkazového riadku aplikácie rovnako ako vstupný súbor a použitú databázu.

3.1.2 Analýza požiadavkov

Požiadavky na aplikáciu z predošlej podkapitoly 3.1.1 je nutné analyzovať a definovať postupy, ktorými su dosiahnuté požadované ciele. Cieľom je získať výstupný súbor obsahujúci mieru zhody vstupných a výstupných dát a určiť presnosť použitej komerčnej geolokačnej databázy.

Formát vstupných a výstupných dát v podobe polí oddelených jednotným oddeľovačom, medzerou alebo tabulátorom, umožňuje použiť viacero techník na rozdelenie do jednotlivých dátových polí v jazyku Python, napríklad pomenovanej n-tice. Dátová štruktúra pomenovaná n-tica je popísaná v [9].

S použitými komerčnými geolokačnými databázami je možné komunikovať dvoma spôsobmi, pomocou webového rozhrania alebo aplikačného rozhrania. U oboch spôsobov je komunikácia realizovaná na úrovni dotazov protokolu `HTTP`, metódami `GET` a `POST`. Pre podmnožinu komerčných geolokačných databáz použitých v tejto práci využívajú metódu `GET` databázy MaxMind [14] a IPInfo [17].

IP adresa hľadanej stanice je v prípade týchto databáz prenášaná rozličnými spôsobmi. V prípade databázy MaxMind [14], je IP adresa prenášaná ako súčasť URL adresy, nie ako jej parameter. Rovnakým spôosobm je prenášaná IP adresa aj u databázy IPInfo [17]. Príklady jednotlivých URL adries sú:

https://www.maxmind.com/geopip/v2.1/city/192.168.1.1?demo=1
https://ipinfo.io/10.4.3.56/json?token=iplocation.net

Metódu `POST` používa zvyšných päť databáz. Tieto databázy predstavujú:

- DB-IP [15].
- IP2Location [16].
- Skyhook [18].
- EurekAPI [19].
- Geobytes [20].

Metódou `POST` sú dáta spolu s IP adresou prenášané v tele dotazu. U každej z databáz je IP adresa prenášaná v rozdielne pomenovanom poli. Dotaz na databázu DB-IP [15] nesie informáciu o IP adrese v poli `address`. Pre databázy EurekAPI [19] a Skyhook [18] je informácia o IP adrese prenášaná v poli `ip`, u databázy IP2Location [16] v poli `ipAddress` a u databázy Geobytes [20] v poli `fqcn`.

Odpoveďou na jednotlivé dotazy sú dáta v podobe webovej stránky v jazyku HTML alebo dáta vo formáte JSON [22]. Pre prístup k vybraným sekciám kódu v jazyku HTML sú využívané regulárne výrazy a možnosti modulu alebo inač povedané knižnice *re*. V prípade formátu JSON [22] je výber informácií jednoduchší, použitím modulu *json*.

Po extrahovaní týchto dát je možné prejsť k operáciam nahradenia alebo odrezania defivovaných v pravidlách, vytvorených skriptov popísaným v podkapitole 3.2. Porovnávanie dát sa uskutoční na úrovni znakov jednotlivých reťazcov a výsledkom je už spomínaný reťazec zhody, ktorý nadobúda hodnoty *YES*, *NO* alebo *UNK*.

Vzľadom na fakt, že viaceré geolokačné služby komerčných databáz sú platené, poskytujú databázy možnosť otestovania si vlastností a schopností týchto služieb prostredníctom bezplatnej registrácie na obmedzenú dobu alebo s akceptovaním limitov pre odosielané dotazy. V prípade, že aplikácia pri spracovaní narazí na niektorý zo spomenutých obmedzení, pokračuje vo vyhodnocovaní ale namiesto získaných dát z geolokačných databáz vypisuje za vstupnými dátami a výstupným poľom identifikujúcim použitú databázu reťazec upozorňujúci na túto udalosť.

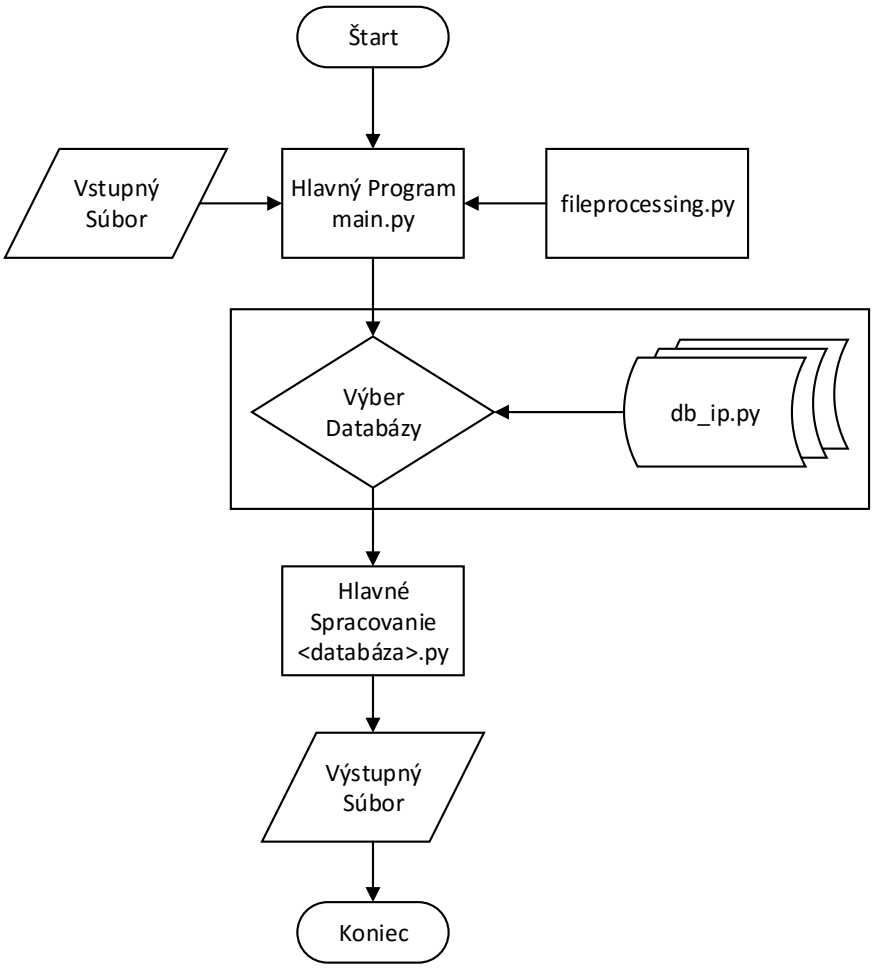
Všetky spomenuté výsledky sú následne zapísané do výstupného súboru v predpísanom formáte a oddelené oddeľovačom voleným užívateľom. Takto pripravené dáta sú predmetom ďalšej podrobnej analýzy pre vyhodnotenie presnosti použitej geolokačnej databázy. Po tejto analýze požiadavkov nasleduje návrh aplikácie a jej kľúčových časti.

3.1.3 Návrh aplikácie

Po špecifikácii a analýze požiadavkov aplikácie nasleduje fáza návrhu. Táto fáza je dôležitá najmä pre možnosti rozšírenia a udržovateľnosti kódu aplikácie. Dobrým návrhom aplikácie je zaistené, aby pri akomkoľvek požiadavku na rozšírenie funkcionality existujúceho kódu nedošlo k výraznému vplyvu tejto zmeny na celkový proces spracovania aplikácie. Táto zmena musí byť koncovému užívateľovi skrytá,

aby mohol nerušene využívať rozhranie aplikácie bez nutnosti prispôsobovania sa.

Pre potrebu možnosti rozšírenia funkcionality aplikácie a schopnosti prispôsobenia sa okolitým zmenám bola zvolená modulárna štruktúra. Tá umožňuje rozdeliť jednotlivé procesné úkony do samostatných modulov, ktoré navzájom komunikujú prostredníctvom definovaných rozhraní. Rozhrania týchto modulov sú tvorené funkciami. Úlohy jednotlivých modulov zobrazuje diagram na obrázku 3.1. Hlavný mo-



Obr. 3.1: Vývojový diagram aplikácie na spracovanie geolokačných informácií

dul, `main.py`, začína importovaním jednotlivých modulov štandardnej knižnice jazyka Python a pokračuje importovaním vlastnými modulmi. Prvým z vlastných modulov je importovaný modul poskytujúci funkcie pre spracovanie vstupného súboru s testovanými geolokačnými záznamami, modul `fileprocessing.py`. Nasledujúcimi importovanými modulmi sú moduly testovaných komerčných geolokačných databáz. Po importovaní všetkých modulov dochádza ku spracovaniu parametrov príkazového riadku, ktorými je definované výsledné chovanie programu.

V špecifikácií požiadavkov na aplikáciu v podkapitole 3.1.1, boli rozoberané viaceré možnosti voľby parametrov programu. Medzi tie patrí:

- Parameter nápovedy `-h`.
- Parameter logovacieho výstupu `-v`.
- Parameter výberu oddeľovača polí vstupných záznamov `-i`.
- Parameter výberu oddeľovača polí výstupných záznamov `-o`.
- Parameter výberu databázy `-d`.
- Parameter súboru pravidiel pre nahradenie `-r`.
- Parameter súboru pravidiel pre odrezanie `-c`.

Jediným nespomenutým v zozname parametrov je povinný parameter názvu vstupného súboru, ktorý obsahuje vstupné geolokačné záznamy. Po spracovaní parameterov príkazového riadku programu dôjde k načítaniu riadkov vstupného súboru a ich uloženiu do dátovej štruktúry, ktorá je predávaná modulom databáz prostredníctvom rozhraní definovaných funkciami.

Po spracovaní vstupných záznamov dôjde následne k výberu modulu, ktorý zodpovedá výberu testovanej komerčnej geolokačnej databázy na základe parametra príkazového riadku. Tomuto modulu sú predané dáta v dátovej štruktúre, ktorá je výsledkom spracovania vstupného súboru. Modulárna štruktúra umožňuje rozšíriť testovanú podmnožinu geolokačných databáz pridaním ďalšieho modulu k programu. Podstatou je výber daného modulu databázy na základe parametru príkazového riadku a dátovej štruktúry slovníka, ktorá odkazuje na jednotlivé moduly názvom databázy ako indexom. Nevýhodou zvoleného prístupu je nutnosť modifikácie kódu pre pridanie ďalšieho modulu na viacerých miestach a to:

- Importovaním nového modulu príkazom `import`.
- Pridaním položky do slovníkovej dátovej štruktúry.
- Upravením parametrov príkazového riadku.
- Upravením nápovedy príkazového riadku.

Ihneď po importovaní vybraných modulov geolokačných databáz dochádza k naplneniu dátovej štruktúry slovníka odkazmi na jednotlivé moduly. Zvolením jedného alebo všetkých modulov v závislosti na výbere hodnoty parametra príkazového riadku sa vykoná hlavné spracovanie v konkrétnom databázovom module. To je definované funkciou `check_ips()` rovnakou pre každý databázový modul a s rovnakými parametrami.

Týmito parametrami sú:

- Dátová štruktúra obsahujúca vstupné geolokačné záznamy.
- Oddeľovač polí výstupných záznamov.
- Názov súboru pre pravidlá nahradzovania.
- Názov súboru pre pravidlá odrezávania.
- Zapnutie alebo vypnutie výpisu logu aplikácie.

Zavedením rovnakej hlavičky funkcie predstavujúcej spracovanie databázou zaisťuje odolnosť k vnútorným zmenám implementácie jednotlivých modulov napríklad v dôsledku zmeny webového alebo aplikačného rozhrania použitej databázy. Táto vlasnosť tiež umožňuje naviazať aplikáciu na ďalšie spracovanie externými prostriedkami, akými sú napríklad ďalšie programy.

Po získaní geolokačných informácií z databáz a následným zapísaním výstupných dát do výstupného súboru končí hlavné spracovanie. Výstupný súbor je tak pripravený na ďalšie spracovanie či analýzu. Prvou analýzou môže byť kontrola zhody jednotlivých polí za účelom tvorby pravidiel, ktorá je predmetom podkapitoly 3.2.

3.1.4 Implementácia

Táto podkapitola rozoberá implementáciu jednotlivých modulov a špecifických častí plikácie navrhnutej v predchádzajúcej časti. Prvým zo spomínaných modulov je hlavný modul `main.py`. Tento modul predstavuje vstupný bod celej aplikácie pre získavanie geolokačných záznamov z komerčných geolokačných databáz.

Hlavný modul `main.py` na začiatku testuje verziu intepretu, s ktorou bola aplikácia spustená a spracováva argumenty príkazového riadku štandardným modulom jazyka Python, modulom `argparse`. Po uložení hodnôt argumentov príkazového riadku, nasleduje vytvorenie adresára pre uloženie výstupných súborov. Nasleduje definícia funkcie `main()`, ktorá spúšťa hlavný cyklus spracovania. Táto funkcia je zavolaná na konci hlavného modulu.

V úvode tejto funkcie dochádza k spracovaniu vstupného súboru a načítaniu jeho obsahu do dátovej štruktúry poľa, ktoré obsahuje ďalšiu dátovú štruktúru známu ako pomenovaná n-tica. Spracovanie vstupného súboru sa deje v module `fileprocessing.py`, ktorý okrem definície funkcie `get_ip_records()` definuje aj dátovú štruktúru `_IP_RECORD`. Táto štruktúra predstavuje pomenovanú n-ticu obsahujúcu všetky polia záznamu vo vstupnom súbore. Pole týchto štruktúr je vrátené funkciou `get_ip_records()` do hlavnej funkcie modulu `main.py`. Dátová štruktúra je zobrazená vo výpise 3.1.

Výpis 3.1: Dátová štruktúra uchovávajúca vstupný geolokačný záznam

```
_IP_RECORD = collections.namedtuple("_IP_RECORD",   "id \
ip \
dns \
continent \
countryCoordinate \
regionCoordinate \
cityCoordinate \
unknown_parameter_1 \
unknown_parameter_2 \
latitudeCoordinate \
longitudeCoordinate \
dns_correction \
row \
correct")
```

Pre oddelenie polí vstupných záznamov je použitý ďalší zo štandardných modulov jazyka Python, modul `CSV`. Tento modul umožňuje definovať oddeľovač polí vstupných záznamov a rozdeliť ich na základe tohto údaju. Po získaní všetkých polí jednotlivých záznamov, sú tieto polia vložené do dátovej štruktúry `_IP_RECORD` a táto štruktúra je uložená do poľa obsahujúceho všetky záznamy zo vstupného súboru. Po vrátení tohto poľa funkciou `get_ip_records()` naspäť, do funkcie `main()`, v hlavnom module aplikácie je možné prejsť k testovaniu databáz. Dôležitou časťou funkcie `main()` je cyklus, ktorý v prípade, že je hodnota argumentu pre použitú databázu `all`, predstavujúca všetky dostupné moduly databáz, prechádza slovníkovú dátovu štruktúru a volá spracovanie každého databázového modulu prostredníctvom funkcie `check_ips()`. Výpis 3.2 ukazuje časť kódu funkcie `main()`, ktorý sa zameriava na tento cyklus.

Výpis 3.2: Testovanie všetkých databáz

```
def main():
  ...
  if arguments.database == "all":
    for value in databases.values():
      value.check_ips(ip_records,
              arguments.output_separator,
              arguments.cut,
              arguments.replace,
              arguments.verbose)
  ...
```

Ako vidieť vo výpise 3.2, slovník obsahujúci odkazy na moduly jednotlivých databáz je pomenovaný `databases`. Nad získaným odkazom na modul v premennej `value` je volaná funkcia `check_ips`, ktorá príjíma ako argument pole vstupných záznamov získaných po spracovaní modulom `fileprocessing.py`. Ďalšími argumentami tejto funkcie sú oddeľovač polí výstupných záznamov, názov súborov s pravidlami pre odrezávanie a nahradzovanie a posledným parametrom je parameter pre voľbu zobrazenia dodatočných informácií pri spracovaní.

Implementácia funkcie `check_ips()` sa v každom databázovom module odlišuje v niekoľkých bodoch. Prvým bodom je výber metódy protokolu `HTTP`, ktorou komunikuje modul s databázou. V podkapitole 3.1.2, kde bola rozoberaná analýza požiadavkov, bolo spomenuté využitie metódy `GET` u databáz MaxMind [14] a IPInfo [17]. Ako už bolo ukázané, u databázy MaxMind [14] je dotazovaná IP adresa umiestená v URL adrese. IP adresa je získavaná z predanej dátovej štruktúry obsahujúcej záznamy zo vstupného súboru. Všetky záznamy sú prechádzané v cykle, rovnakom pre všetky moduly a funkcie `check_ips()` a odkazovaná IP adresa je pridávaná do URL adresy spájaním reťazcov ako vidieť vo výpise 3.3.

Výpis 3.3: Vytváranie URL adresy pre dotaz GET na databázu MaxMind

```
for ipRecord in ipRecords:
  ...
  url = "https://www.maxmind.com/geoip/v2.1/city/" +
      ipRecord.ip + "?demo=1"
  ...
```

Takto zostavená URL adresa je následne použitá pri odosielaní dotazu metódou `urllib.request.urlopen(url)`. U metódy `POST` je situácia odlišná. Pre zostavenie dotazu na databázu, napríklad Skyhook [18], je nutné vyplniť a zakódovať viaceré polia prenášané v tele dotazu. Tieto polia sú zobrazené vo Výpise 3.4.

Výpis 3.4: Polia dotazu POST na databázu Skyhook

```
parameters = {
  'version': '2.0',
  'ip': ipRecord.ip,
  'prettyPrint': 'true',
  'key': '<Auth Key>',
  'user': 'eval'
}
```

Následne sú tieto polia zakódované metódou `urlencode()`. Takto pripravené polia sú predané ako parameter konštruktoru objektu `HTTP` dotazu a odoslané metódou `urlopen(request)`.

Ďalšou z odlišností implementácie u jednotlivých modulov je spôsob, akým je spracovávaná odpoveď z databázy. Vzhľadom na rozdielne služby sú dáta často zakódované v odlišnom tvare avšak za podmienok zachovania formátu sa jedná o HTML dokument alebo dokument vo formáte JSON. V prípade HTML dokumentu je nutné lokalizovať sekcie kódu zodpovedajúce geolokačným dátam pre dotazovanú IP adresu. To bolo vykonané počas implementácie a vyústilo k zostaveniu viacerých regulárnych výrazov pre vyhľadávanie konkrétnych častí u databázy IP2Location [16].

Hlavným regulárnym výrazom je výraz `html_table`, ktorý je spolu s ďalšími regulárnymi výrazmi preložený a uložený do premenných na začiatku kódu modulu. Tento regulárny výraz predstavuje tabuľkové dáta z HTML dokumentu vráteného ako odpoveď na dotaz na polohu danej stanice prostredníctvom IP adresy. Jednotlivé riadky tabuľky sú pridané do premennej `table_rows` ako výsledok zhody s prijatým HTML dokumentom. Následne sú jednotlivé riadky prechádzané v cykle a aplikovaním ďalších regulárnych výrazov `html_location` a `html_lat_lon` sú vyhľadávané konkrétne údaje o polohe dotazovanej stanice. Časť cyklu s hľadaním údajov o krajine, mieste, regióne, zemepisnej šírke a dĺžke je vo Výpise 3.5.

Po spracovaní odpovede regulárnymi výrazmi sú jednotlivé sekcie kódu HTML obsahujúce informácie o polohe stanice uložené do premenných pre ďalšie spracovanie, ktorým môže byť napríklad aplikovanie pravidiel pre odrezanie častí reťazcov alebo nahradenie. Tieto operácie sú predmetom podkapitoly 3.2 avšak súčasťou každého z modulov databáz.

Výpis 3.5: Spracovanie odpovede od databázy IP2Location regulárnymi výrazmi

```
response_data = response.read().decode('utf-8')
...
table_rows = html_table.findall(response_data)
...
if table_rows is not None:
  for row in table_rows:
    if row[0] == "Location":
      location = html_location.search(row[1])
      if location is not None:
        country_estimation = location.group(1).strip()
        region_estimation = location.group(2).strip()
        city_estimation = location.group(3).strip()
      ...
    if row[0] == "Latitude & Longitude":
      lat_lon = html_lat_lon.search(row[1])
      if lat_lon is not None:
        latitude_estimation = float(lat_lon.group(1))
        longitude_estimation = float(lat_lon.group(2))
      ...
  ...
```

Ak je odpoveďou dokument vo formáte JSON [22], spracovanie je jednoduchšie. Príkladom odpovede vo formáte JSON je databáza Skyhook [18]. Získaná odpoveď po odoslaní `HTTP` dotazu je dekódovaná a predaná ako parameter metódy štandardného modulu jazyk Python *json*, metóde `json.loads()`. Táto metóda prevedie reťazec znakov získaných v odpovedi do dátovej štruktúry slovníka. Slovníková štruktúra následne umožňuje použitie vlastných metód pre manipuláciu s dátami a jednoduchý prístup k nim prostredníctvom hranatých zátvoriek. Príklad spracovania odpovede vo formáte demonštruje kód vo Výpise 3.6.

Výpis 3.6: Spracovanie odpovede od databázy Skyhook vo formáte JSON

```
content = response.read().decode('utf-8')
content_json = json.loads(content)
...
if content_json.get('data'):
  if content_json['data'].get('civic'):

    # Country Information
    if content_json['data']['civic'].get('countryIso'):
      country = content_json['data']
                ['civic'].get('countryIso')
...
```

Nad použitou slovníkovou štruktúrou je volaná jej vstavaná metóda `get()`, ktorá prijíma parameter v podobe reťazca, ktorý predstavuje kľúč. Ak sa kľúč nachádza v danej časti slovníkovej štruktúry, vracia odpoveď v podobe hodnoty, ktorá sa ukrýva za ním. V opačnom prípade vráti hodnotu `None`. Takýmto spôsobom je spracovávaná každá časť odpovede vo formáte JSON pre získanie hodnôt polí odkazujúcich sa na krajinu, región, mesto, zemepisnú šírku a dĺžku.

Táto podkapitola uzatvára proces vývoja aplikácie pre získavanie geolokačných informácií z komerčných geolokačných databáz. Zameriavala sa predovšetkým na kľúčové aspekty vývoja, analýzu a špecifikáciu požiadavok a časť implementácie kľúčových častí modulov celej aplikácie. Nasledujúca podkapitola 3.2 sa zameriava na viackrát spomenuté pravidĺa, ktoré umožňujú zvýšiť na prvý pohľad malú presnosť niektorých z testovaných databáz a tak prispieť k lepšiemu vzájomnému porovnávaniu.

3.2 Úprava geolokačných záznamov

V predchádzajúcej podkapitole 3.1 boli spomenuté právidlá pre nahradzovanie a odrezávanie reťazcov v geolokačných údajoch získaných z databáz. Tieto údaje sú obsiahnuté v poliach vo výstupnom súbore za reťazcom definujúcim použitú databázu. Jedná sa predovšetkým o polia definujúce:

- Krajinu.
- Región.
- Mesto.

U niektorých geolokačných údajov sa môže stať, že sú pred porovnávaním so správnou polohou stanice definovanou v odpovedajúcom poli vstupného záznamu v nesprávnom formáte, čo zapríčiňuje výsledok nezhody. Tento formát môže predstavovať významovo rovnakú informáciu zapísanú odlišnou sekvenciou znakov a tým, že sú reťazce týchto polí porovnávané znak po znaku dôjde k spomínanej nezhode. Na základe tohto nedostatku môže dôjsť k nesprávnemu štatistickému odhadu presnosti testovanej geolokačnej databázy. Riešením tohto problému je zavedenie pravidiel, ktoré upravia výstupné polia s geolokačnými údajmi do formátu odpovedajúcemu vstupným dátam dotazovanej stanice čím v zaručia, že dôjde požadovanej a skutočnej zhode.

3.2.1 Vytváranie pravidiel

Špecifikácie obidvoch pravidiel hovoria o vytvorení dvoch súborov, ktoré budú obsahovať zvlášť pravidlá pre odrezávanie a zvlášť pre nahradzovanie reťazcov v poliach obsajúcich geolokačné údaje získane z geolokačných databáz. Pravidlá sú vytvárané na základe analýzy súboru s geolokačnými záznamami, ktorý je výstupom aplikácie popisovanej v predchádzajúcej podkapitole 3.1.

Pre účely tejto analýzy bol vytvorený skript `analyzer.py`. Analýza súboru obsahujúceho získané geolokačné záznamy z testovanej databázy prebieha na základe voľby parametra príkazového riadku `-p`, ktorý prijíma hodnoty `YES`, v prípade užívateľ chce prechádzdať všetky polia záznamov a vytvárať pravidlá alebo `NO` v opačnom prípade. Ďalším povinným parametrom je názov vstupného súboru.

Analýza tohto súboru prebieha na základe prechádzania polí záznamov definujúcich zhodu alebo nezhodu na úrovni krajiny, regiónu alebo mesta. Po nájdení reťazca identifikujúceho nezhodu danej úrovne ako `NO` je možné pristúpiť k vytváraniu pravidiel. Na výber z možností sú pravidla pre odrezávanie z reťazcov alebo pravidlá pre nahradzovanie reťazcov v geolokačných údajoch získaných z databázy.Príkladom je Výpis 3.7, obsahujúci časť kódu spracovania hlavného cyklu v skripte `analyzer.py`.

Výpis 3.7: Spracovanie poľa označujúceho nezhodu na úrovni krajiny

```
if record[14] == 'NO':
  country_no += 1
  if arguments.perform == 'yes':
    if pairs.get(record[13]) is None:
      print("old:<"+record[4]+">\nnew:<"+record[13]+">\n")
      choice = input("... create Replace rule? y/n")
      if choice == 'y':
        operation('r',replace_dict,record[4],record[13])
        country_rules += 1
      choice = input("... create Cut rule? y/n")
      if choice == 'y':
        operation('c', cut_dict, record[4], record[13])
        country_rules += 1
      pairs[record[13]] = []
      pairs[record[13]].append(record[4])
    else:
      if not (record[4] in pairs[record[13]]):
        ... Creating rules ...
        ...
        pairs[record[13]].append(record[4])
else:
  if record[14] != 'UNK':
    country_yes += 1
```

Premenná `record` obsahuje všetky polia jedného záznamu vo vstupnom súbore. Ten je prechádzaný v cykle a jednotlivé riadky záznamov sú načítané do tejto premennej. Jednotlivé indexy odpovedajú konkretným geolokačným údajom alebo reťazcom definujúcim zhodu na danej úrovni. Index 14 odpovedná poľu, ktoré obsahuje informáciu o zhode na úrovni krajiny. Po úspešnej podmienke testovania na nezhodu daného poľa je možné pristúpiť k ďalšej analýze za účelom vytvorenia pravidiel. Predtým sa však počítadlo nezhodujúcich sa dát na úrovni krajiny zvýši o jedna. Dôvodom pre existenciu tohoto počítadla je vytvorenie správy obsahujúcej pomer správnych a nesprávnych odhadov polôh na rôznych úrovniach.

Po preskúmaní hodnoty argumentu `-p`, uloženej v premennej `perform`, možno prejsť k vytváraniu pravidiel alebo preskočiť túto časť za účelom získania len konečnej správy o počte správnych a nesprávnych odhadov. V prípade kladnej hodnoty `yes` dôjde ku kontrole, či prechádzaný geolokačný údaj už nebol predmetom skúmania za účelom vytvorenia pravidiel.

Pri prechádzani jednotlivých polí obsahujúcich daný geolokačný údaj je nutné udžiavať informáciu, či tento údaj nebol už predmetom skúmania. Pre tieto účely slúži premenná `pairs`, ktorá predstavuje slovníkovú dátovú štruktúra. Indexami slovníku sú geolokačné údaje získane z databázy, ktoré boli konfrontované v dôsledku nezhody. Hodnotou na danom indexe je pole, ktoré obsahuje všetky vstupné geolokačné údaje stanice porovnávanej úrovne. V prípade, že sa opakovane vyskytne rovnaký pár vstupných a výstupných geolokačńých údajov, predpokladá sa, že pravidlo pre tento pár už bolo vytvorené alebo bolo zamietnunté. Toto testovanie vykonáva podmienka vo Výpise 3.7, ktorá testuje pomocou slovníkovej metódy `get()` výskyt indexu výstupného poľa.

Špeciálny prípad nastane, ak výstupný geolokačný údaj bol predmetom skúmania, avšak vstupný údaj je odlišný. V takomto prípade môže dôjsť k vytváraniu pravidiel a vstupný geolokačný údaj sa vloží do poľa vstupných údajov pod indexom výstupného geolokačného údaju.

Pravidlá sa vytvárajú prostredníctvom funkcie `operation`, ktorá príjima štyri parametre. Prvým z parametrov je identifikátor operácie, ktorý môže nadobúdať hodnoty `'r'` pre operáciu nahradzovania a hodnotu `'c'` predstavujúcu odrezávanie. Na základe tejto hodnoty vykonáva telo funkcie odlišné operácie. Nahradzovací identifikátor spúšťa vytváranie pravidla vo formáte dvoch reťazcov, nahradzovaným a nahradzovacím, oddelených znakom tabulátora. Toto pravidlo je po vytvorení zapísané do súboru, ktorý tvorí druhý parameter funkcie `operation`. Ďalšie dva parametre sú využívané pri vytváraní nahradzovacieho pravidla a obsahuju pôvodné vstupné a výstupné geolokačné údaje.

Po prejdení všetkých polí záznamov vo vstupnom súbore je vygenerovaná správa, predstavovaná súborom obsahujúcim reťazec *report* v názve. Obsahom tejto správy je pomer správnych a nesprávnych odhadov a počty vytvorených pravidiel pre jednotlivé úrovne. Takto vytvorené pravidlá môžu byť aplikované na testovanú vzorku a opätovne vyhodnotené skriptom `analyzer.py` spusteným s parametrom `p` nastaveným na hodnotu `no` pre okamžitú správu o náraste presnosti po použiti vytvorených pravidiel.

3.2.2 Aplikovanie pravidiel

Vytvorené pravidlá možno aplikovať znovu na testovanú vzorku a docieliť tak spomínané zlepšenie štatistických výslekov pojednávajúcich o presnosti odhadu geolokačných databáz. Aplikovanie týchto pravidiel je vykonávané v aplikácií pre získávanie geolokačných záznamov, v tele funkcie `check_ips()` prostredníctvom vnútorne definovanej funkcie `operation()`. Táto funkcia má rovnaký názov ako v prípade funkcie vytvárajúcej pravidla v aplikácií `analyzer.py` avšak narozdiel od vytvárania pravidiel ich aplikuje na získané geolokačné údaje. Funkcia je zobrazená vo Výpise 3.8

Výpis 3.8: Funkcia pre aplikáciu pravidiel

```
def operation(optype, string, file, input_record):
  if optype == 'replace':
    replace_dict = open(file, 'r', encoding='utf-8')
    reader = csv.reader(replace_dict, delimiter='\t')
    for record in reader:
      if record[0] in string:
        old = string
        string =
      string.replace(record[0],record[1]).strip()
        if verbose:
          print('ROW: ' + ' '.join(input_record.row) +
          ', OPERATION: REPLACE, BEFORE:' +
          old + ' AFTER: ' + string)
    replace_dict.close()

  if optype == 'cut':
    cut_dict = open(file, 'r', encoding='utf-8')
    for record in cut_dict:
      record = record.strip()
      if record in string:
        old = string
        string = string.replace(record, '').strip()
        if verbose:
          print('ROW: ' + ' '.join(input_record.row) +
          ',OPERATION:CUT,BEFORE:'+old+' AFTER: '+string)
    cut_dict.close()
  return string
```

Po získaní odpovede z databázy v hlavnom spracovaní funkcie `check_ips()` a extrahovaní potrebných geolokačných údajov je možné prístúpiť k aplikácií pravidiel. Predtým sa však testuje, či boli prítomné parametre `-r` a `-c` pri spustení aplikácie na získavanie geolokačných údajov. Tieto parametre obsahujú hodnoty názvov súborov s pravidlami pre nahradzovanie a odrezávanie.

V prípade existencie spomenutých parametrov príkazovej riadky a tiež názvov jednotlivých súborov, je vykonaná aplikácia pravidiel na získané geolokačné údaje. Zavolaním funkcie `operation()` sú postupne spracovávané geolokačné údaje rôznych úrovni.

Funkcia `operation()` je volaná s niekoľkými parametrami. Prvým parametrom je identifikovaná zamýšlaná operácia. Môže sa jednať o reťazec `replace` pre aplikovanie pravidiel nahradzovania alebo o reťazec `cut` pre aplikovanie pravidiel odrezávania. V oboch prípadoch sú pravidlá aplikované na reťazec v druhom parametri funkcie. Tretím parametrom je názov súboru obsahujúci pravidlá.

Ak je reťazcom prvého parametra funkcie, reťazec `replace`, prechádazajú sa jednotlivé pravidlá prostredníctvom metód už spomenutého modulu *CSV*. Dôvodom výberu tohoto modulu je formát pravidiel, ktorý predstavujú reťazce oddelené jednotným znakom tabulátora. Ak nahradzovaný alebo nahradzovací reťazec obsahuje medzeru, nedôjde k nesprávnemu načítaniu hodnôt. Po načítaní oboch reťazcov pravidiel je následne testovaný výskyt prvého, nahradzované reťazca v geolokačnom údaji. Pred aplikovaním nahradzovacieho reťazca je uložená predošlá hodnota geoloakačného za účelom zápisu logu programu pri prítomnosti parametra `verbose` v nadradenej funkcii `check_ips()`. Pri operácií odrezávania sa spôsob aplikovania pravidla líší v jeho načítaní zo súboru jednoduchým iterovaním cez všetky riadky daného súboru a testovanie výskytu niektorého z nich v geolokačnm údaji predanom v druhom parametri funkcie `operation()`.

Týmto je uzatvorená Kapitola 3, pojednávajúca o tvorbe aplikácií pre praktickú časť tejto práce. Nasleduje Kapitola 4, ktorá aplikuje vytvorené aplikácie na podmnožinu testovaných komerčných geolokačných databáz za účelom hodnotenia presnosti odhadov polôh testovaných staníc.

4 TESTOVANIE

Pred vyslovením záveru o presnosti geolokačných služieb testovaných komerčných geolokačných databáz je nutné vykonať testovanie na vybranej vzorke staníc s IP adresami. Vzhľadom na určité obmedzenia pri využití bezplatných skúšobných verzií geolokačných služieb je dôležité prispôsobiť rozsah testovanej vzorky. V tejto práci boli použité testovacie dáta predstavujúce 400 záznamov staníc so známou polohou, ktoré tvorili vstup aplikácie na získavanie geolokačných údajov.

V nasledujúcich podkapitolách bude vysvetlený postup testovania jednotlivých komerčných geolokačných databáz spolu s ich obmedzeniami. Testovanie zahrňuje použitie aplikácie na získavanie geolokačných údajov a tiež aplikácie na tvorbu pravidiel pre opakované testy.

4.1 Databáza DB-IP

Prvou z testovaných databáz bola databáza DB-IP. Komunikácia s touto databázou je vykonávaná prostredníctvom jej webového rozhrania, ktoré obsahuje formulár so vstupným poľom pre IP adresu dotazovanej stanice. Cez toto rozhranie boli posielané dotazy a vyhodnocované údaje, spracovávané v niekoľkých fázach. Obmedzením tohto prístupu bolo odoslanie maximálne 200 dotazov na databázu za deň. Pre tieto účely bola testovaná vzorka záznamov rozdelená do dvoch skupin po 200 záznamoch.

Po prvom testovaní možno vyhodnotiť presnosť vo vybraných úrovniach nasledujúcim spôsobom. Presnosť na úrovni krajiny zaznamenala nulovú zhodu na testovanej vzorke. Oproti tomu presnosť odhadu na úrovni regiónu zaznamenala úspešné 4 záznamy zo 400, čo predstavuje 1% z celkového počtu testovaných záznamov. Presnosť na úrovni mesta zaznamenala zhodu v 158 prípadoch.

Po vytvorení približne 40 pravidiel a ich následnej aplikácií pri opakovanom teste, došlo k výraznému nárastu presnosti na úrovni krajiny na úroveň 375 zhodných údajov, kde možno hovoriť o 93.75% poklese nezhodujúcich sa údajov. V prípade zhody na úrovni regiónu možno hovoriť o 69.23% náraste oproti pôvodnej hodnote. Počet správnych odhadov na úrovni mesta sa zvýšil o 22.78% oproti pôvodnej hodnote.

4.2 Databáza EurekAPI

Databáza EurekAPI neobmedzuje počet dotazov za deň alebo za iný časový údaj. Napriek tomu je nutné sa vopred zaregistrovať a obdržať autentizančný kľúč, ktorý je použitý pri odosielaní každého dotazu. Overením je odpoveď obsahujúca geolokačné údaje dotazovanej stanice vo formáte JSON.

Kľúč je nutné vložiť do zdrojového kódu modulu pre databázu EurekAPI. Je identifikovaný premennou `key`. Výpis 4.1 zobrazuje úsek kódu modulu pre databázu EurekAPI, kde je nutné vytvorený autentizačný kľuč vložiť.

Výpis 4.1: Umiestnenie autentizačného kľúča pre databázu EurekAPI

```
################ KEY ############################
key = 'XXXXXXXXXXXXXXX'                  #AUTH KEY!#
#################################################
```

Po vložení kľúča je možné začať s testovaním. Po prvom testovaní vzorky 400 staníc bez použitia pravidiel na odrezávanie a nahradzovanie, boli získané nasledujúce výsledky. Presnosť na úrovni krajiny zaznamenala zhodu v 374 prípadoch. Pri testovaní sa vyskytli aj výsledky s neznámou hodnotou krajiny. Presnosť na úrovni regiónu obsahovala zhodu v 3 prípadoch a presnosť na úrovni mesta zhodu v 164 prípadoch.

Vytvorením 21 pravidiel pre nahradzovanie došlo k zvýšeniu počtu správnych odhadov polohy na úrovni regiónu, kde bol nárast až o 300% oproti pôvodnej hodnote a v prípade miest to bol 12.8% nárast.

4.3 Databáza Geobytes

Databáza Geobytes rovnako ako databáza EurekAPI neobmedzuje počet prijatých dotazov z jednej verejnej IP adresy za jednotku času ani nevyžaduje registráciu a následné overenie identity registrovaného užívateľa. Preto je možné prístupiť k testovaniu celkovej vzorky 400 staníc bez obmedzení.

Prvé testovanie prinieslo zhodu na úrovni krajiny v 333 záznamoch. Úroveň regiónu zaznamenala úspešný odhad pri 4 záznamoch a úroveň mesta pri 140 záznamoch. Po vytvorení 29 pravidiel došlo k nárastu presnosti ná úrovni krajiny o 3%, na úrovni regiónu o 275% a na úrovni mesta o 10.71%.

4.4 Databáza IP2Location

Testovanie databázy IP2Location patrilo medzi najkomplikovanejšie úlohy. Vzhľadom na obmezdenie prijatých dotazov za deň na 50, bolo nutné vytvoriť 8 vzoriek dát po 50 záznamoch, aby bolo možné získať výsledky celkovej testovanej zložky 400 staníc.

Prvým testovaním bola dosiahnutá presnosť na úrovni krajiny správnym odhadom v 4 záznamoch, na úrovni regiónu v 7 záznamoch a na úrovni mesta v 134 záznamoch. Aplikovaním 65 vytvorených pravidiel došlo k zvýšeniu presnosti odhadu na úrovni krajiny zo 4 pôvodných správnych odhadov na 374. Na úrovni regiónu došlo k 171.42% nárastu správnych odhadov a na úrovni mesta k zvýšeniu o 33.58%.

4.5 Databáza IPInfo

Rovnako ako v prípade databáz Geobytes či EurekAPI, ani databáza IPInfo neobmedzuje počet prijatých dotazov za časovú jednotku alebo interval. Nie je tiež nutné vykonať registráciu užívateľa za účelom testovania geolokačnej služby. Pre tieto vlastnosti je opäť možné testovať celkovú testovaciu zložku 400 staníc bez obmedzení.

Prvé výsledky testovania priniesli mieru zhody na úrovni krajiny v 366 prípadoch, na úrovni regíonu v 4 a na úrovni mesta v 149 prípadoch. Po vytvorení a následnej aplikácií pravidiel došlo k zvýšeniu počtu správnych odhadov na úrovni regiónu o 50% a na úrovni mesta o 14.09%.

4.6 Databáza MaxMind

Databáza MaxMind poskytuje použitie demo verzie pre testovanie presnosti jej geolokačných služieb. Táto demo verzia však obsahuje obmedzenie vo forme približne 90 dotazov za deň. V tomto prípade bolo opäť nutné vytvoriť niekoľko testovaných vzoriek aby bolo dosiahnuté pokrytie všetkých testovaných 400 staníc.

Výsledky testov v prvej fáze bez aplikovania pravidiel vykazujú zhodu na úrovni krajiny v 374 prípadoch a na úrovni mesta v 167 prípadoch. Počet zhodných záznamov na úrovni regiónu bol v tomto prípade nulový. Vytvorením 19 právidiel došlo k zvýšeniu počtu správnych odhadov o 18.56% na úrovni miest a na úrovni regiónu stúpol počet správnych odhadov z 0 na 5.

4.7 Databáza Skyhook

Poslednou testovanou databázou je databáza Skyhook, ktorá rovnako nezavádza žiadne obmedzenia pri jej testovaní. Počas prvej fázy testovania celkovej zložky 400 staníc je vidieť výrazný nedostatok tejto databázy predstavujúci chýbajúce geolokačné údaje u prevažnej väčšiny. Na úrovni krajiny sa jedná o 275 neznámych hodnôt, na úrovni regiónu 329 a na úrovni mesta až 334 nezámych hodnôt. Dôsledkom tohto stavu nemožno aplikovať vytvorené pravidlá na tieto hodnoty a preto je nárast zhody na rôznych úrovniach zanedbateľný.

Vytvorením 13 pravidiel došlo k nulovému navýšeniu počtu správnych odhadov na úrovni krajiny pri aktuálnej hodnote 123. Ná úrovni regiónu došlo k zmene počtu správnych odhadov z hodnoty 0 na hodnotu 2 a na úrovni mesta došlo k navýšeniu o 16.21% oproti pôvodnej hodnote.

Testovanie databázy Skyhook uzatvára Kapitolu 4. V Kapitole 5 budú zhrnuté výsledky testov a bude vyhodnotená miera presnosti na rôznych úrovniach u jednotlivých komerčných geolokačných databáz.

5 ANALÝZA VÝSLEDKOV TESTOVANIA

Po dôslednom testovaní komerčných geolokačných databáz, nasleduje vyhodnotenie výsledkov a určenie presnosti. Vzhľadom na limity, ktoré niektoré databázy pri testovaní stanovujú, budu výsledky analýzy prezentované na testovanej vzorke s počtom 400 staníc. Táto vzorka je dostatnočná pre približné určenie kvality jednotlivých geolokačných služieb komerčných geolokačných databáz.

Výsledky testov boli vo výraznej miere odlišne pred a po použití pravidiel na úpravy výstupných geolokačných údajov, čo je možné vidieť na tabuľkách 5.1 a 5.2. Pre aplikakovaním týchto pravidiel nereflektovali namerané výsledky skutočný stav, preto rozdiely na rôznych úrovniach polohy danej stanice sú značne odlišné.

Tab. 5.1: Výsledky testov pred použitím pravidiel

Databázy	**Vstupné záznamy (400)**								
	Krajina			**Región**			**Mesto**		
	Zhoda	**Nezhoda**	**Neznáme**	**Zhoda**	**Nezhoda**	**Neznáme**	**Zhoda**	**Nezhoda**	**Neznáme**
DB-IP	0	400	0	4	396	0	158	242	0
EurekAPI	374	14	12	3	299	98	164	139	97
Geobytes	333	57	10	4	386	10	140	250	10
IP2Location	4	391	5	7	388	5	134	261	5
IPInfo	366	24	10	4	275	121	149	131	120
MaxMind	374	14	12	0	306	94	167	142	91
Skyhook	123	2	275	0	71	329	37	29	334

Tab. 5.2: Výsledky testov po použití pravidiel

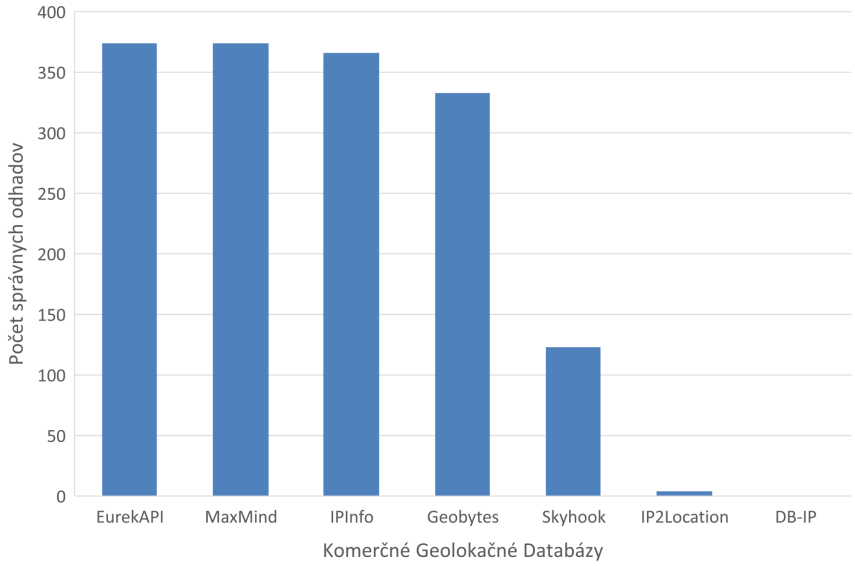
Databázy	**Vstupné záznamy (400)**								
	Krajina			**Región**			**Mesto**		
	Zhoda	**Nezhoda**	**Neznáme**	**Zhoda**	**Nezhoda**	**Neznáme**	**Zhoda**	**Nezhoda**	**Neznáme**
DB-IP	375	25	0	13	387	0	194	206	0
EurekAPI	374	14	12	14	288	98	185	118	97
Geobytes	343	47	10	15	375	10	155	235	10
IP2Location	374	21	5	19	376	5	179	216	5
IPInfo	366	24	10	6	273	121	170	110	120
MaxMind	374	14	12	5	301	94	198	111	91
Skyhook	123	2	275	2	69	329	43	23	334

Zo zobrazených tabuliek je jasne vidieť nárast počtu zhody na rôznych úrovniach u viacerých databáz. Použitie pravidiel takto demonštruje svoje opodstatnenie a ukazuje presnejšie výsledky testov odrážajúcich kvalitu použitých komerčných geolokačných databáz. Miera zhody na jednotlivých úrovniach je predmetom nasledujúcich podkapitol.

5.1 Presnosť na úrovni krajiny

Zo spominaných úrovni geolokačných údajov je miera zhody na úrovni krajiny najúspešnejším parametrom pri hodnotení presnosti jednotlivých databáz. Vzhľadom na rozsiahlu geografickú oblasť, ktorú krajiny vo väčšine prípadov pokrývajú, možno považovať za najjednoduchšie získavaný geolokačný údaj. Preto poloha staníc v rámci krajiny predstavuje najväčšiu mieru zhody u všetkých databáz s výnimkou databázy Skyhook, ktorá obsahuje veľke množstvo chýbajúcich údajov.

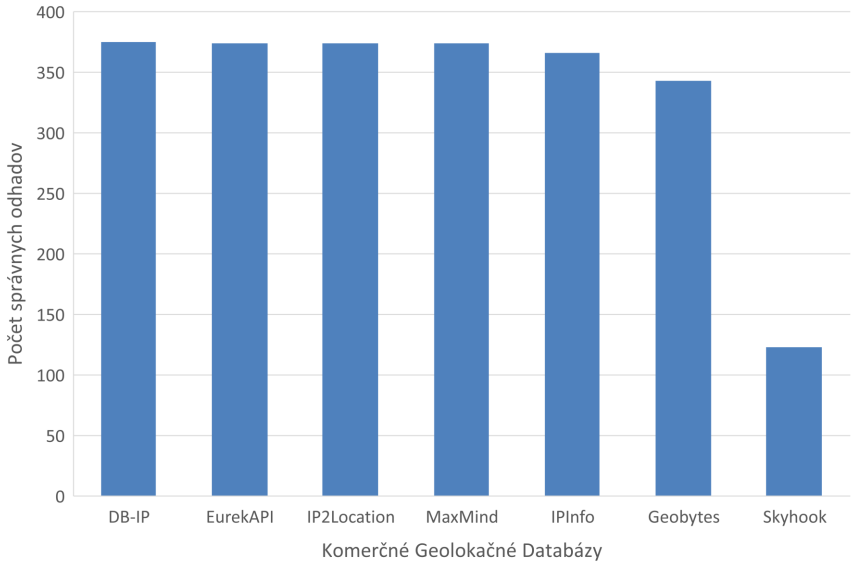
Pre použitím pravidiel vykazuje najvyššiu mieru zhody na úrovni krajiny databáza EurekAPI spolu s databázou MaxMind. Za nimi nasledujú ostatné databázy spolu s databázou DB-IP, ktorá ma tejto úrovni nulovú zhodu. Dôvodom tejto nulovej hodnoty je to, že databáza narozdiel od ISO kódu danej krajiny, vracia geolokačné údaje v bežnom slovnom tvare a tak nedôjde k zhode v žiadnom z testovaných záznamov. Podobná situácia sa vyskytuje aj u databázy IP2Location, ktorá rovnako zobrazuje geolokačné údaje krajiny v slovnom formáte. Výsledky zhody na úrovni krajiny zobrazuje histogram na obrázku 5.1.



Obr. 5.1: Počet správnych odhadov krajiny pred použitím pravidiel

Po aplikovaní pravidiel vytvorených analýzou prvotných dát, sa databázá DB-IP dostáva na vrchol hodnotenia presnosti na úrovni krajiny. Rovnako sa databáza IP2Location zaradila mezdi prvé štyri, na úrovni krajiny najpresnejšie databázy. Rozdiel na výsledných priečkach je zanedbateľný. Spomedzi všekých testovaných databáz májú na odhad krajiny danej stanice najlepšie výsledky databázy DB-IP, IP2Location, EurekAPI a databáza MaxMind.

Na obrázku 5.2 je zobrazený histogram po aplikovaní pravidiel. Najlepšie výsledky dosahuje DB-IP. V ďalších úrovniach sa očakáva dominancia v počte správnych odhadov práve tejto databázy.

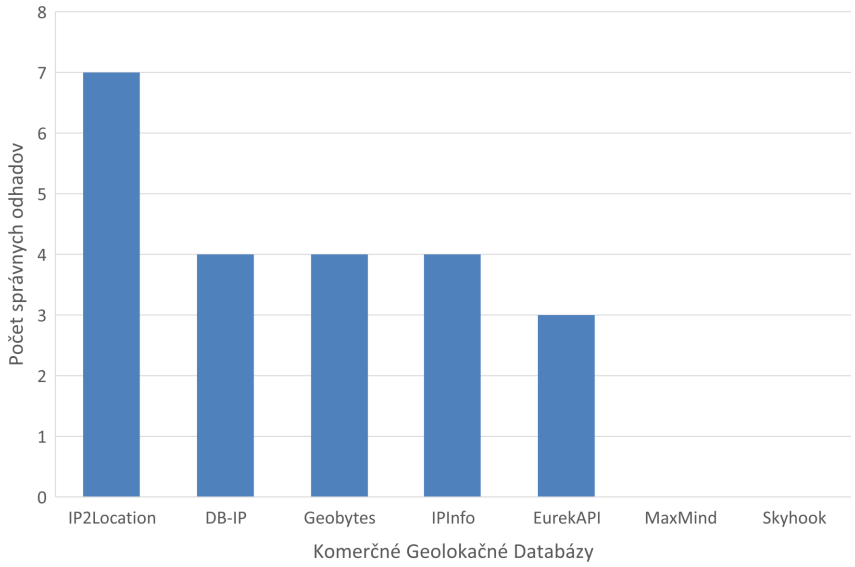


Obr. 5.2: Počet správnych odhadov krajiny po použiti pravidiel

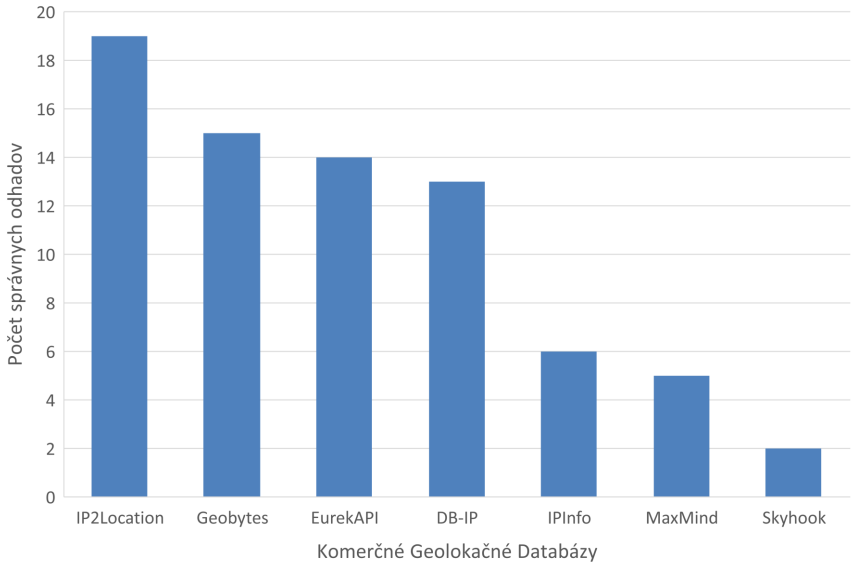
5.2 Presnosť na úrovni regiónu

Situácia pri prenosti na úrovni regiónu je u komerčných geolokačných databáz najkomplikovanejšia. Vzhľadom na fakt, že žiadna z testovaných databáz nepoužíva štandardizovaný formát zápisu regiónu je úroveň zhody výsledkom porovnávania reťazcov znak po znaku, ktorá je často aj po aplikovaní pravidiel nedostatočná. Príkladom je histogram zobrazený na obrázku 5.3, kde je vidieť, že maximálny počet správne odhadnutých regiónov je 8, čo predstavuje 2% z celkového počtu testovaných staníc. Histogram ukazuje počet správne odhadnutých regiónov pred aplikovaním pravidiel pre nahradzovanie a odrezávanie.

Výsledok po aplikácií pravidiel zobrazuje histogram na obrázku 5.4. Aj napriek použitiu pravidiel nestúpol počet úspešných odhadov regiónov natoľko aby bolo možné jednoznačne tvrdiť, že databáza IP2Location je na tejto úrovni najpresnejšia.



Obr. 5.3: Počet správnych odhadov regiónu pred použitím pravidiel

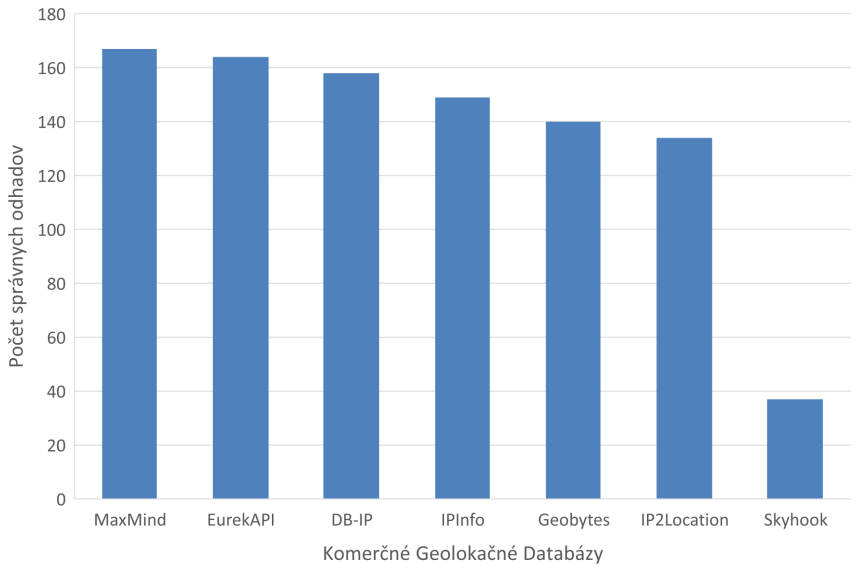


Obr. 5.4: Počet správnych odhadov regiónu po použití pravidiel

5.3 Presnosť na úrovni miest

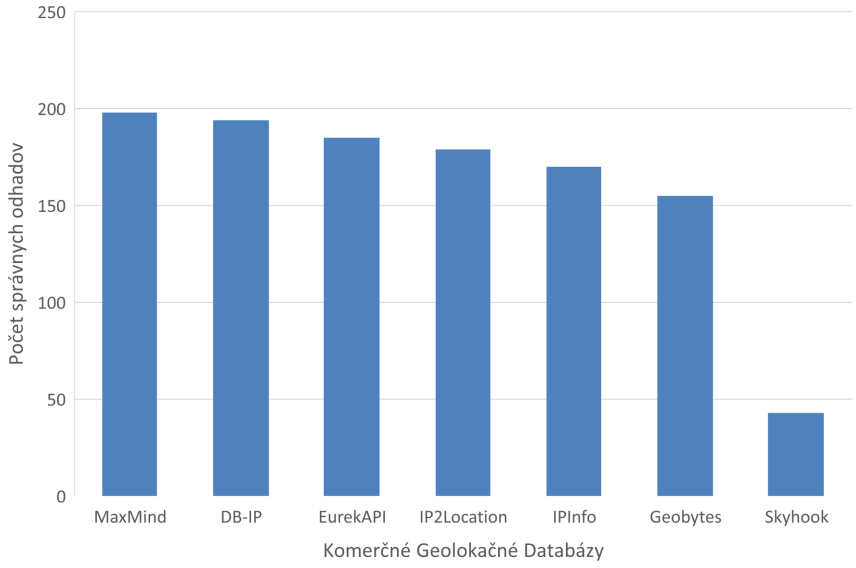
Poslednou sledovanou úrovňou odkazujúcou sa na geogragickú oblasť tvoriacu jeden územný celok je úroveň miest. Navzdory nízkej miere zhody v predošlej úrovni regiónu, je počet správnych odhadov miest neporovnateľne vyšší. Dôvodom tohto je väčšia unikátnosť názvov jednotlivých miest ako aj globálna identifikácia prevažne anglickým názvom. Aj napriek týmto vlastnostiam sa však môže vyskytnúť prípad, kedy je nutné vytvoriť pravidlo pre dôkaz skutočného stavu zhody na úrovni mesta.

Histogram na obrázku 5.5 zobrazuje najlepšiu mieru zhody geolokačných údajov na úrovni mesta u databázy MaxMind.



Obr. 5.5: Počet správnych odhadov mesta pred použitím pravidiel

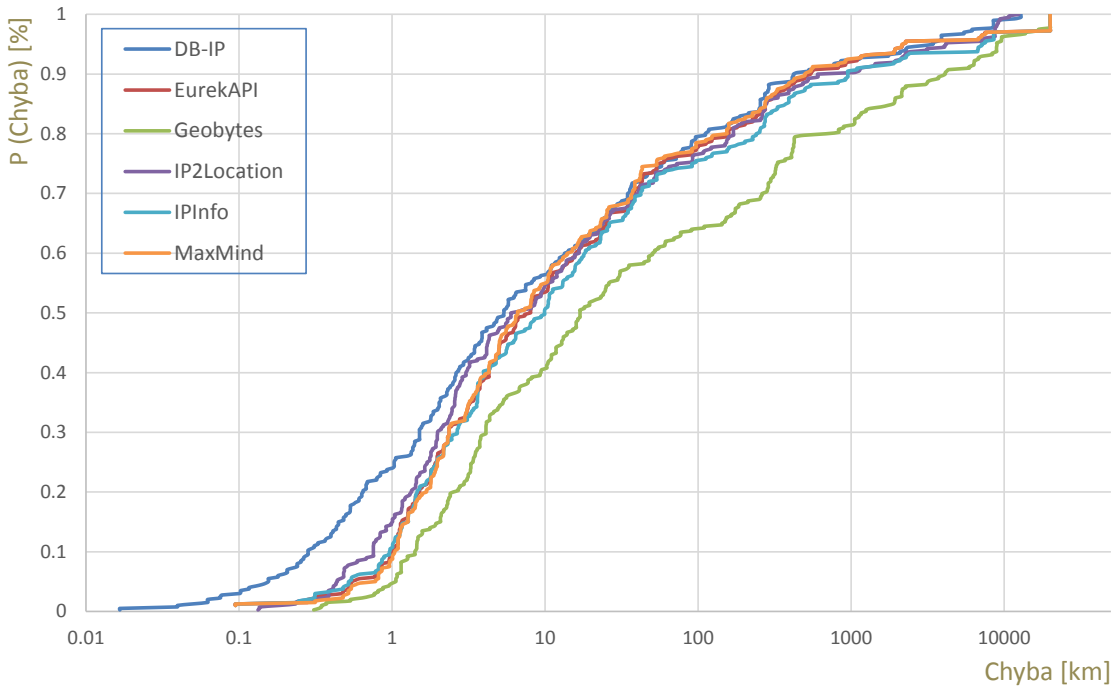
Aplikovanie pravidiel opäť pôsobí pozitívne na databázu DB-IP, ktorá preskočí počtom správnych odhadov databázu EurekAPI a zaradí sa na drúhe miesto pri počte správnych odhadov na úrovni miest. Tento stav demonštruje histogram na obrázku 5.6.



Obr. 5.6: Počet správnych odhadov mesta po použití pravidiel

5.4 Chyba odhadu

Posledným rozoberaným parametrom určujúcim mieru presnosti testovaných geolokačných databázi je chyba odhadu vypočítana ako vzdialenosť v kilometroch. Vytvorením kumulatívnej distribučnej funkcie pravdepodobnosti výskytu chyby danej veľkosti umožňuje nahliadnúť na presnosť geolokačných služieb v inom merítku. Tendencia rastu jednotlivých kriviek zastupujúcich rôzne geolokačné databázy predstavuje lepšiu a horšiu presnosť odhadu presnej polohy. Na obrázku 5.7 je zobrazená kumulatívna distribučná funkcia všetkých databáz s výnimkou databázy Skyhook, ktorá, ako už bolo popisované, počas fázy testovania obsahovala veľke množstvo chýbajúcich geolokačných údajov jednotlivých staníc.



Obr. 5.7: Kumulatívna distribučná funkcia pre testované databázy

Presnosť geolokačných databáz možno určiť na základe tendencie rastu kriviek kumulatívnych distribučných funkcií v smere osy y, ktorá reprezentuje už spomínanú pravdepodobnosť výskytu chyby danej veľkosti. Spomedzi všetkých testovaných databáz má v tomto smere najrýchlejší rast databáza DB-IP, ktorú je možné týmto označiť za najpresnejšiu z databáz na testovanej vzorke 400 staníc. Znovu je nutné zdôrazniť, že veľkosť testovanej vzorky nemusí byť vždy dostačujúca na konečné hodnotenie presnosti všetkých testovaných komerčných geolokačných databáz. Napriek rozsahu testovanej vzorky a prvých priečkach tejto databázy pri odhadoch na úrovni krajím a miest, je možné ohodnotiť databázu DB-IP vzhľadom na rovnaké podmienky ako najpresnejšiu.

6 VYHODNOTENIE ANALÝZY

Po testovaní a analýze získaných výsledkov je možné prístupiť k vyhodnoteniu presností testovaných komerčných geolokačných databáz. Testovanie sa uskutočnílo na rovnakej vzorke 400 staníc za rovnakých podmienok behu aplikácií a pre každú z databáz prinieslo výsledky určujúce jej presnosť na rôznych úrovniach definujúcich geografické celky akými sú napríklad krajíny, regióny a mestá. Okrem týchto úrovni bola určená presnosť odhadov skutočnej polohy stanice na základe jej zemepisných súradnic výpočtom chyby v kilometroch. Táto vzdialenostná chyba poslúžila ako základ pre vytvorenie kumulatívnej distribučnej funkcie, ktorá poukazovala na pravdepodobnosť výskytu chýb rôznej veľkosti.

Spomedzi všetkých testovaných komerčných geolokačných databáz vykazuje najlepšiu mieru presnosti databáza DB-IP. Okrem najvyššej dosiahnutej presnosti na úrovni krajiny dosiahla rovnako dobré výsledky aj pri presnosti odhadu na úrovni miest. Zrychlený nárast krivky v smere osy y na grafe kumulatívnej distribučnej funkcie podtrháva fakt, že spomedzi všetkých testovaných komerčných geolokačných databáz dosahuje najlepšie výsledky vo výskytoch chyby pod jeden kilometer, čo môže predstavovať prijateľnú hodnotu chyby vzdialenosti od skutočnej polohy hľadanej stanice.

Ako už bolo spomenuté, testovacia vzorka 400 staníc sa nemusí javiť ako dostačujúca pre konečný dôkaz o presnosti použitej komerčnej geolokačnej databázy. Napriek tejto nevýhode je možné vysloviť záver, že spomedzi testovaných geolokačných databáz vykazujú akceptovateľné výsledky pre túto jednoduchú analýzu na úrovní krajín, miest a zemepisných súradníc všetky až na databázu Skyhook. Tá vykazovala veľké množstvo chýbajúcich geolokačných dát testovaných staníc. Okrem databázy Skyhook ostatné databázy preukázali použiteľnosť v odchadoch polohy staníc s rozdielom úrovne regiónu, ktorá vzhľadom na svoju nejednoznačnú reprezentáciu vratenú z geolokačných databáz nebola akceptovateľná pre úvahu o presnosti.

7 ZÁVER

Diplomová práca sa zaoberala analýzou komerčných geolokačných databáz. Tieto databázy poskytujú svoje geolokačné služby za poplatky spojené s garanciou presnosti na rôznych úrovniach. Medzie tieto úrovne patrí správny odhad krajiny, regiónu, mesta, zemepisnej dĺžky a šírky. Na vzorke testovacích dát bola vyhodnotená ich presnosť. Cieľom tejto práce bolo vykonať podrobnú analýzu za týmto účelom a vyjadriť stanovisko k existujúcim možnostiam IP Geolokácie.

Úvodná časť práce bola venovaná teoretickému základu o IP Geolokaćii, existujúcich technikách a dostupných službách komerčných geolokačných databáz. Prístup k existujúcim geolokačným záznamom je hlavnou doménou týchto databáz a predstavuje pasívny prístup k IP Geolokácií. Komunikácia s týmito databázami prebieha prostredníctvom dvoch typov rozhraní, webovým a aplikačným.

V náväznosti na teoretické základy IP Geolokácie bolo možné pristúpiť k návrhu aplikácií pracujúcich s komerčnými geolokačnými databázami a spracovaním získaných geolokačných dát. Komunikácia prebieha na úrovni protokolu HTTP a odosielaní jeho dotazov. Tie obsahujú informáciu o IP adrese dotazovanej stanice. Odpoveďou komerčných geolokačných databáz je webová stránka vo formáte HTML alebo dokument vo formáte JSON.

Modulárnym návrhom aplikácie na získavanie geolokačných záznamov je možné rozšíriť testovanú podmnožinu komerčných geolokačných databáz a tým lepšie zmapovať súčasný stav dostupných geolokačných služieb.

Pre potreby automatizácie procesu získavania geolokačných dát a ich následnej analýze bolo nutné naštudovať možnosti programovacieho jazyka Python. Ten umožňuje vytvárať požiadavky protokolu HTTP prostredníctvom metód modulu *urllib*, ktorý je súčasťou jeho štandardnej knižnice. Geolokačné dáta zakódované v odpovediach od komerčných geolokačných databáz boli získavané použitím metód štandardných modulov jazyka Python, modulov *re* a *json*.

Zo získaných dát bolo možné vytvoriť súbory pravidiel pre získánie presnejších údajov o miere zhody v jednotlivých úrovniach medzi ktoré patria úroveň krajiny, regiónu a miest. Opakovaným testom a použitím vytvorených pravidiel s následnou konfrontáciou s predošlých výsledkov, bolo možné určiť skutočnú hodnotu presnosti použitej komerčnej geolokačnej databázy. Táto presnosť predstavuje počet zhodných dát na spomenutých úrovniach.

Výsledkom analýzy nameraných dát bolo možné rozhodnúť o presnosti jednotlivých komerčných geolokačných databáz nielen na úrovni krajín, regiónov a miesta ale aj jednoznačným určením vzdialenostnej odchýlky súradnicovej polohy hľadanej stanice. Napriek malej testovanej vzorke staníc, je možné použiť výsledky analýzy komerčných geolokačných databáz pre predstavu o ich možnostiach v rámci odhadu polohy na úrovni krajín aleb miest a vybrať tak tú, ktorá najviac vyhovuje potrebám využitia. Presnosť odhadu na rôznych úrovniach nie je vždy jediným parametrom výberu komerčnej geolokačnej databázy. Dôležitým parametrom je aj cena danej geolokačnej služby predstavujúcou garanciu presnosti. Práve pre tento nezanedbateľný parameter je nutné vykonávať vlastnú analýzu, pre porovnanie skutočných a na trhu prezentovaných presností.

Vyhodnotením výsledkov analalýzy bolo dospené k záveru, že na testovanej vzorke staníc bola najpresnejšia databáza DB-IP, ktorá vykazovala najlepšie výsledky na úrovniach krajín a miest. Táto databáza prezentuje svoje možností ako najrozsiahlejšie pokrytie IPv4 a IPv6 adresových priestorov spolu s najvyššiou presnosťou dostupnou na trhu. Spomedzi zvyšných testovaných komerčných geolokačných databáz dopadla v testoch najmenej presvedčivo databáza Skyhook, ktorá obsahovala veľkú mieru chýbajúcich údajov.

Ciele tejto práce boli splnené a výsledna analýza dostačuje na vyslovenie záveru o presnosti testovaných komerčných geolokačných databáz na testovanej vzorke staníc so známou polohou. S neustále meniacou sa infraštruktúrou Internetu sa však nedá s určitosťou povedať, že databázy ktoré v súčasnosti vykazujú slabé a nie vždy dostačujúce výsledky nebudú v budúcnosti zastupovať najlepšie riešenia na trhu dostupých geolokačných služieb.

LITERATÚRA

[1] PUŽMANOVÁ, R. *TCP/IP v Kostce*. 2. vyd. Kopp, 2009. 620 s. ISBN: 978-80-7232-388-3.

[2] *Linux: dokumentační projekt.* 4., aktualiz. vyd. Brno: Computer Press, 2007, 1334 s. ISBN 978080-251-1525.

[3] POESE, I., UHLIG, S. KAAFAR, M., DONNET, B., GUEYE, B. IP Geolocation Databases: Unreliable? *SIGCOMM Computer Communication Review*, 2011, roč. 41, č. 2, s. 53-56. ISSN: 0146-4833

[4] DONG, Ziqian, Rohan D.W. PERERA, Rajarathnam CHANDRAMOULI a K.P. SUBBALAKSHMI. Network measurement based modeling and optimization for IP geolocation. *Computer Networks* [online]. 2012, 56(1): 85-98 [cit. 2015-12-11]. DOI: 10.1016/j.comnet.2011.08.011. ISSN 13891286. Dostupné z URL: <`http://linkinghub.elsevier.com/retrieve/pii/S1389128611003173`>.

[5] ERIKSSON, Brian, et al. A learning-based approach for IP geolocation. *Passive and Active Measurement* Springer Berlin Heidelberg, 2010. p. 171-180.

[6] KOCH, Robert; GOLLING, Mario; RODOSEK, Gabi Dreo. Advanced Geolocation of IP Addresses. *International Conference on Communication and Network Security (ICCNS)*. 2013. p. 1-10.

[7] WANG, Ting; SONG, Junde; SONG, Meina. An Optimization Method for the Geolocation Databases of Internet Hosts based on Machine Learning. *Mathematical Problems in Engineering*. 2015. p. 1-17

[8] CASTRO, Elizabeth a Bruce HYSLOP. *HTML5 a CSS3: názorný průvodce tvorbou WWW stránek*. 1. vyd. Brno: Computer Press, 2012, 439 s. ISBN 978-80-251-3733-8.

[9] SUMMERFIELD, Mark. *Python 3: výukový kurz*. Vyd. 1. Brno: Computer Press, 2010, 584 s. ISBN 978-80-251-2737-7.

[10] POSTEL, J. *Internet Control Message Protocol* [online]. c1981. [cit. 10. 12. 2015]. Dostupné z URL: <`https://tools.ietf.org/html/rfc792`>.

[11] REKHTER, Y., Li, T. a HARES, S. *A Border Gateway Protocol 4 (BGP-4)* [online]. c2006. [cit. 10. 12. 2015]. Dostupné z URL: <`https://www.ietf.org/rfc/rfc4271.txt`>.

[12] STEPHENS, Ryan K, Ronald R PLEW a Arie JONES. *Naučte se SQL za 28 dní: [stačí hodina denně].* Vyd. 1. Brno: Computer Press, 2010, 728 s. ISBN 978-80-251-2700-1.

[13] MAZIKU, Hellen, et al. Enhancing the classification accuracy of IP geolocation. *MILITARY COMMUNICATIONS CONFERENCE, 2012-MILCOM 2012.* IEEE, 2012. p. 1-6.

[14] MaxMind *IP Geolocation and Online Fraud Prevention* [online]. [cit. 10. 12. 2015]. Dostupné z URL: <`https://www.maxmind.com/en/home`>

[15] DB-IP *DB-IP - IP Geolocation and Network Intelligence* [online]. [cit. 10. 12. 2015]. Dostupné z URL: <`https://db-ip.com/`>

[16] IP2Location *IP Address Geolocation to Identify Website Visitor's Geographical Location* [online]. [cit. 10. 12. 2015]. Dostupné z URL: <`http://www.ip2location.com/`>

[17] IPInfo *IP Address Details - ipinfo.io* [online]. [cit. 10. 12. 2015]. Dostupné z URL: <`http://ipinfo.io/`>

[18] SkyHook *Skyhook | Global Location and Context Software Products* [online]. [cit. 10. 12. 2015]. Dostupné z URL: <`http://www.skyhookwireless.com/`>

[19] EurekAPI *IP Address Geolocation to trace Country, Region, City, ZIP Code, etc* [online]. [cit. 10. 12. 2015]. Dostupné z URL: <`https://www.eurekapi.com/`>

[20] Geobytes *Get City Details* [online]. [cit. 10. 12. 2015]. Dostupné z URL: <`http://geobytes.com/get-city-details-api/`>

[21] FIELDING, R., GETTYS, J., MOGUL, J., FRYSTYK, H., MASINTER, L., LEACH, P., Berners-Lee, T. *Hypertext Transfer Protocol – HTTP/1.1* [online]. c1999. [cit. 10. 12. 2015]. Dostupné z URL: <`https://tools.ietf.org/html/rfc2616`>.

[22] BRAY, T., Google, Inc. *The JavaScript Object Notation (JSON) Data Interchange Format* [online]. c2014. [cit. 10. 12. 2015]. Dostupné z URL: <`https://tools.ietf.org/html/rfc7159`>.

[23] FIELDING, Roy Thomas. *Architectural styles and the design of network-based software architectures.* 2000. PhD Thesis. University of California, Irvine.

[24] KANISOVÁ, Hana a Miroslav MÜLLER. *UML srozumitelně.* 2., aktualiz. vyd. Brno: Computer Press, 2006. ISBN 80-251-1083-4.

ZOZNAM SYMBOLOV, VELIČÍN A SKRATIEK

ISP	Poskytovaľ internetových služieb – Internet Service Provider
LAN	Lokálna sieť – Local Area Network
IP	Internetový protokol – Internet protocol
ISO	Mezdinárodná Organizácia pre Štandardizáciu – International Organization for Standardization
HTTP	Hypertextový prenosový protokol – Hypertext Transfer Protocol
HTML	Hypertextový značkovací jazyk – Hypertext Markup Language
JSON	Javascriptový objektový zápis – Javascript Object Notation

ZOZNAM PRÍLOH

A OBSAH PRILOŽENÉHO CD

Obsahom priloženého CD sú zdrojové súbory, elektronická verzia textu práce, výsledky textov a analýzy komerčných geolokačných databáz. Jednotlivé úrovne adresárovej štruktúry obsahuju `readme.txt` súbory s nápovedou o obsahu priloženého CD a práce s aplikáciami. Adresárová štruktúra je nasledujúca.

- Text/
  - `analyza geolokacnych databaz.pdf`
- Vysledky Analyzy/
  - `Analyza Geolokacnych Databaz.xlsx`
- Vysledky Testovania/
  - `Vysledky Testovania.zip`
- Zdrojove Subory/
  - Ziskavanie Geolokacnych Dat/
    - `db_ip.py`
    - `eurek.py`
    - `fileprocessing.py`
    - `geobytes.py`
    - `ip2location.py`
    - `ipinfo.py`
    - `main.py`
    - `maxmind.py`
    - `neustar.py`
    - `skyhook.py`
    - `readme.txt`
  - Uprava Geolokacnych Dat/
    - `analyzer.py`
    - `readme.txt`
- `input.dat`
- `readme.txt`